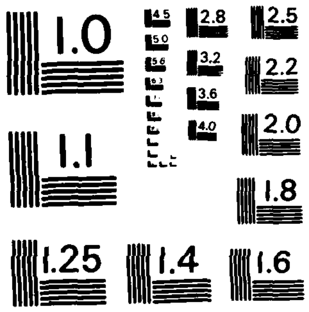
AD-A127 754   A PRODUCTIVITY ANALYSIS OF NONPROCEDURAL LANGUAGES(U)   1/2
NAVAL POSTGRADUATE SCHOOL MONTEREY CA
M CORCORAN ET AL. DEC 82

UNCLASSIFIED   F/G 9/2   NL


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6

2

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

A PRODUCTIVITY ANALYSIS
OF NONPROCEDURAL LANGUAGES

by

MIMI CORCORAN
DENHAM B. MACMILLAN
December 1982

Thesis Advisor: Norman Lyons

Approved for Public Release; Distribution Unlimited

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A127 754	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) A Productivity Analysis of Nonprocedural Languages		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis December, 1982
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Mimi Corcoran Denham B. Macmillan		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		12. REPORT DATE December, 1982
		13. NUMBER OF PAGES 116
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for Public Release; Distribution Unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Nonprocedural languages, fourth generation languages, query languages, application languages, nonprogrammers

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The emergence of so-called "nonprocedural" languages promises the elimination of many of the problems encountered in managing information systems, as well as increasing productivity, by offering a flexible, easy to learn, user friendly language to interact with the host language. This thesis investigates nonprocedural languages in general, with particular attention paid to the languages FOCUS and RAMIS II, in order to ascertain the benefits (Continued)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

ABSTRACT (Continued) Block # 20

and drawbacks of these languages, assess the fulfillment of vendor claims, examine their investment viability, and explore user satisfaction.

A Productivity Analysis of Nonprocedural Languages

by

Mimi Corcoran
Lieutenant, United States Navy
B.S., Pennsylvania State University, 1972

and

Denham B. MacMillan
Lieutenant, United States Navy
B.A., University of South Carolina, 1974

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN INFORMATION SYSTEMS

from the

NAVAL POSTGRADUATE SCHOOL
December 1982

Authors: M Corcoran

Denham B. MacMillan

Approved by: Norman R Lyons
Thesis Advisor

[illegible]
Second Reader

[illegible]
Chairman, Department of Administrative Sciences

[illegible]
Dean of Information and Policy Sciences

ABSTRACT

The emergence of so-called "nonprocedural" languages promises the elimination of many of the problems encountered in managing information systems, as well as increasing productivity, by offering a flexible, easy to learn, user friendly language to interact with the host language. This thesis investigates nonprocedrual languages in general, with particular attention paid to the languages FOCUS and RAMIS II, in order to ascertain the benefits and drawbacks of these languages, assess the fulfillment of vendor claims, examine their investment viability, and explore user satisfaction.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

I. INTRODUCTION

In all areas of industry in both the private and public sectors, the value of information systems has been realized. Management Information Systems and information systems applications have today become an integral part of even the smallest organizations and offer many opportunities to improve managerial effectiveness, operational efficiency, and ultimately productivity. By productivity the reference is to the ability to produce, to effect or bring about production, and to effect increases in value or profit. To industry this could refer to increasing its output, its profits, or both. To the individual it means optimization of his personal attributes and skills.

The different types of users of these Management Information Systems (MIS) vary as much as the information and applications they use. Users range from clerical staff personnel to top-level management. Some require simple retrieval facilities, some users require an ad-hoc inquiry and/or reporting capability, others require vast analytical capabilities, while still others desire the ability to prototype and build models. Today's users attempt to make use of all resources available to them to support their decisions and necessarily increase their productivity. As the need for information has become more acute, and the requests for data processing services and applications have become more frequent, time has become the limiting resource most effecting this productivity.

The time of the user is vital in that he sees himself as the center of all transactions. The "time" he has to wait for information is wasted or lost productivity. Not obtaining the information "on-time" results in poor decisions which ultimately decrease productivity. In his efforts to obtain complete, accurate, valid, and timely information and services, many times the user simply does not have the programming skill to produce the results he wants, and the "application backlog" in the organization's data processing division or branch prevents timely processing of his request. As an example....

> A user requires an application to analyze a set of data and produce a report to stress certain marketing failures to be presented at tomorrow's meeting with the company's Board of Directors. He is a non-programmer and must submit a request to the data processing department to achieve the desired results. Due to the Data Processing Department's application backlog, the desired application will not be scheduled for another 2 years. Even if the application is given priority and pushed to the top of the programming list, the program will still have to be designed, written, coded, run, and the results examined to see if the desired results have been achieved.

Another area which is directly effected is the organization's Data Processing (DP) Department--the programmers and analysts that are trying to meet the organization's existing DP requirements and reduce the application backlog which is part of those requirements. Their time is valuable also, but due to the situation which exists in many organizations, they will spend the majority of their time producing small repetitious programs which could be accomplished by a nonprogrammer with minimal experience and the applications backlog will continue to grow. The story continues....

> When questioned by the head of the DP department about the apparent lack of progress on important projects,

the analyst responded that every time progress was being made, some other small yet time consuming task would be given priority. As a matter of fact, the demand on the DP department was presently so great that other analysts and himself were having to program rather than analyze just to keep their heads above water.

It is a fact that the demand for DP applications today greatly outweighs the supply that can be produced by programmers using prescribed procedural/ structured methods. There simply is not enough programmer "time" available because there is not and never will be enough trained programmers to match the growth of applications demand using currently available structured programming techniques. To solve this problem will require increasing user involvement in the creation of applications and increasing programmer productivity by means of programming and applications generating tools. These requirements have contributed to the need for and development of Nonprocedural Programming Languages (NPL) and their related software packages, the subject of this thesis. Before continuing, however, it should be pointed out that the term nonprocedural is not so much a structural description of the language, as it is a phrase established by the producers of these products for marketing purposes.

Through literature research, questionnaire requests, phone conversations, and personal interviews, this paper will examine and clarify the legitimacy of these NPL's. The background of these NPL's will be examined including the causes leading to their development, their evolution, and where they stand today. The paper will include an in depth analysis of two NPLs, RAMIS II and FOCUS, derived from the industrial response of various firms that are presently using those languages. This analysis will cover the companies' initial acquisition considerations, their present usage of the system, and their satisfaction or dissatisfaction with its performance. Items considered in the analysis include implementation, training, learnability, documentation, vendor support, hardware support/conversion, security/access, performance, improvability/user satisfaction, general opinions, and an overall accessment. The paper will conclude with a cost versus benefit analysis and present final remarks about the viability and future of NPLs.

These Non-Procedural Languages will be examined to determine the costs and benefits associated with them, determine whether vendor claims are fulfilled, examine their investment viability, and measure user satisfaction and ultimately productivity enhancement or degradation.

II. BACKGROUND

The rapid advancements the computer field in the last decade has witnessed the substantial decline of most computer costs, particularly hardware, while simultaneously, programming costs have risen dramatically. A natural consequence is the increased attention programming productivity is receiving from management. Programmers, equipment, management and user personnel, and software are some of the many facets of the programming arena which are being explored for improvement feasibility. The advent of data base management systems (DBMS) has focused management's attention on the benefits to be derived from effective utilization of software enhancements.

It has been estimated that the current steady rise in information processing requirements will create the imminent need for a total number of programmers equal to the population of California [Ref. 1: p. 2]. While this is clearly impossible, it is indicative of the need to take a critical look at the productivity of today's programmers and the tools with which they work. The increasing volumes of required information and the resultant increasing number of programmers required by industry, coupled with escalating programmer costs have shed the light on the possibility of freeing programmers from simple and repetitive tasks and to employ nonprogrammers for such processing, thereby freeing programmers to tackle more complex programming.

A new kind of software capability is now called for which has prompted the emergence of English-like languages--languages whose main claim to fame is their easy understandability and quick learnability--providing a prolific tool for the nonprogrammer. This is a very broad field, encompassing a gamut of natural languages, special purpose languages and nonprocedural languages, to name a few. Definitions of several types of these languages follow:

1. Natural languages allow the user the freedom of unlimited syntax in his "conversation" with the computer.
2. Format defined languages, also known as parameterizing languages, utilize a fill-in-the-blank format. This type of language is directed toward the interface, and the analysis of and response to data passing through the interface [Ref. 2: p. 123].
3. Special purpose languages are ones which are designed to satisfy a single objective. The objective might involve the application area, the ease of use for a particular application, or pertain to efficiency of the compiler or the object code.
4. Problem defining languages literally define the problem and may specifically define the desired input and output, but they do not define a method of transformation. There are significant differences between a problem (and its definition), the method (or procedure) to solve it, and the language in which the method is stated.
5. Problem solving languages are those which can be used to specify a complete solution to a problem. This is a relative term which changes as the state of the art changes. All procedure oriented languages are problem solving.
6. Problem describing languages describe the objective in only very general terms, e.g. CALCULATE PAYROLL. All this does is cite, in the most general way, the problem which is to be solved but gives no indication

of its detailed characteristics, let alone how to solve it. [Ref. 3: p. 21-22] These languages have yet to become a reality.

7. Query languages are high level languages oriented towards ad hoc retrieval of data with fast response. They are generally intended to be used by people who are not professional programmers [Ref. 4: p. 7].

There is certainly considerable overlap among these languages, and the absolute distinctions between them is anything but crystal clear. In fact, some languages fall into multiple categories.

This paper will be limited to a discussion of the interesting and powerful branch of English-like languages known as nonprocedural languages.

They have also been called fourth generation languages--the latest addition to the sequence of language generations. See Table I.

TABLE I
Language Generations

First Generation	Machine Language
Second Generation	Assembler Level Languages
Third Generation	Machine Independent Languages
Fourth Generation	Non-procedural Languages

Other terms used are query languages, declarative languages, info-retrieval languages, and end-user languages. Problems exist with all of these terms since none of them is accurate for all the languages; for example, some do not use a data base. The term nonprocedural conjures up some discontent because many of the languages actually contain procedural code; however, the main thrust of the languages is their nonprocedurality, a marketing buzz word used to stress their newness and capabilities. Unlike procedural languages in which the programmer must specify how something is to be done by supplying precisely detailed instructions for every action which is to be accomplished, nonprocedural languages afford the user the luxury of stating only what is to be done, with no concern as to the detailed procedure of how it is done. Although they have thusfar escaped precise definition, and will undoubtedly continue to do so as our concept of "procedurality" changes over time and with advancements in technology, these languages, employed for defining and solving particular classes of problems, can best be described as "nonprocedural."

A distinction between natural languages and nonprocedural languages is best illustrated by example. Natural languages make use of a free phrase structure format with contextual specification. For instance, in the following example it may be argued that the natural language (b) offers an improvement in clarity over an imaginary (procedural or third generation) language version (a):

a.
```
DO I=1 TO 99 BY 2
PRINT I, I**2
END DO
```

b.
```
PRINT ALL THE ODD NUMBERS BETWEEN ONE
AND NINETY NINE AND THEIR SQUARES.
```

Non-procedural languages, on the other hand, use a standardized fixed format for the specification of problems. Consider another example written in a natural language:

```
a.  LIST THE HOURLY FEES OF ALL THE
    LAWYERS WHO LIVE IN NEW JERSEY
```

This same example, written in SEQUEL, a nonprocedural language, would simply be

```
b.  SELECT HOUR-FEE
    FROM LAWYERS
    WHERE HOMESTATE = NEW JERSEY
```

The distinction herein is the use of English-like language to replace more "unnatural" ways of stating problems as contrasted to eliminating sequential instructions specifying procedurality. That is, natural languages are concerned with making queries more like the spoken word; nonprocedural languages are concerned with eliminating specific sequential instructions that lock the computer into a specific logic for solving the problem. While the two are not mutually exclusive by any means, they are definitely not the same concept. Non-procedural is actually a relative term meaning that decreasing numbers of specific sequential steps need be provided by the user as the state of the art improves. The closer the user's approach to stating his problem without specifying the steps for solving it, the more nonprocedural the language is.

The development of this sort of language is of particular practical value and can be extended to almost any field of computer application, including medical, shipping, city planning, accounting/bookkeeping, air-line reservation systems, banking services, etc. Literally scores of nonprocedural languages are now available. The more widely advertised boast an impressive lists of clients, laundry lists of vital features, "bells and whistles" features, and the promise of increased productivity. Inspection of a limited list of presently available nonprocedural languages indicates their abundance and their wide range of applicability. See Table II. Specific features offered are listed in Figure 2.1 [Ref. 5: p. 151-153]. Boolean operators are AND, OR, NOT and NOR. Relational operators are "greater than," "greater than or equal to," "less than," "less than or equal to," "equals," and "does not equal". Set operators are set operations such as JOIN, INTERSECTION, SELECTION, PROJECTION, DIVISION, UNION and DIFFERENCE. Arithmetic operators include PLUS, MINUS, MULTIPLY, DIVIDE, EXPONENTIAL, and the use of parentheses for separation of operators.

Fourth generation applications development systems boast a sophisticated on-line support environment that provides:

1. Menus and help services to coach the inexperienced user,

2. An efficient command language for the experienced developer,

3. Language sensitive editors that streamline the programming process,

4. On-line compilation and execution services to speed development,

5. A flexible printing capability,

6. Report routing and browsing capabilities,

7. Integrated active data/dictionary/directory control, and coordination.

8. Full screen text editor,

9. Utility function commands,

TABLE II

A Sampling of Non-Procedural Languages

ADASCRIPT	Software A.G.
ADF (Applications Development Facility)	IBM
APPLE (Access Path Producing Language)	Northwestern University
ARPL (A Retrieval Process Language)	Bell Telephone Laboratories
DMNIQ (Data Management Inquiry Facility)	Data General
DM-IV (Data Management - IV)	Honeywell
FOCUS	Information Builders
GPLAN (Generalized Planning System)	Purdue University
INQUIRE	Data General
MANAGE/QUERY	Computer Sciences of Australia
MARK IV	Informatics, Inc.
NOMAD	CSS
NUL (Navigational User's Language)	Institut d'informatique, Namur Belguim
QBE (Query-By-Example)	IBM Yorktown Heights Research Laboratory
RAMIS II (Rapid Access Management Information System)	Mathematica
SQUARE (Specifying Queries As Relational Expressions)	IBM San Jose Research Laboratory
SQUIRAL (Smart Query Interface for Relational Algebra)	University of Utah
SYSTEM 2000	MRI Systems Inc.
TDMS (Time Shared Data Management System)	Systems Development Corporation

10. Prompters,

11. Interfaces with multiple DBMS's [Ref. 6: p. 43].
These languages provide a considerable range of file structures, a host language capability and a data management and report generation language facilities. Nonprocedural languages have also simplified communication between the user and the computer, eliminating some of the "red tape" along the way. See Figure 2.2.

Two widely used languages representative of this type are RAMIS II from Mathematica, Inc., and FOCUS from Information Builders, Inc. These two particular languages will be studied in more detail in the following chapters. Clearly, such languages can significantly reduce the complexity and cost of writing applications programs that access the data base, in addition to facilitating access to a data base by non-expert programmers using the language in "stand-alone" mode [Ref. 7: p. 15]. The conceptual view, or data model, need not correspond to the way the data are stored. Three well known data models are the relational model, in which data are assumed to be stored in the form of tables; the hierarchical model, in which data are assumed to be stored in the form of tree structures; and the network model, in which data are assumed to be stored in the form of general graph structures. The choice of a data model provokes controversy among data base designers. The relational model employed imposes little constraint on the way that a user is able to interpret and utilize data. There are no complex tree or network structures that force all users to limit their view of the relationships in the data base to a particular single logical view. The virtue of the relational model is its simplicity and ease of description for a wide variety of users. Experience at Deere & Co. with IBM's

	Boolean Operators	Relational Operators	Set Operators	Arithmetic Operators
ADASCRIPT	all	yes	some	all
APPLE	AND OR NOT	yes	some	none
ARPL	AND OR NOT	yes	some	all
CONVERT	AND OR	yes	some	all
DMINQ	AND OR NOT	yes	some	A & E
DM-IV	no	yes	some	A & E
GPLAN	AND OR NOT	yes	some	all
INQUIRE	no	no	none	none
FOCUS	AND OR NOT	all	some	all
MANAGE/QUERY	AND OR NOT	yes	some	all
MARK IV	AND OR	yes	none	none
NOMAD	AND OR NOT	yes	none	A & E
NUL	AND OR	yes	some	P
QBE	AND OR NOT	yes	all	none
RAMIS II	AND OR NOT	yes	some	all
SEQUEL	AND OR NOT	yes	all	P
SQUIRAL	AND OR	yes	all	none
SQUARE	all	yes	all	A & P
SYSTEM 2000	AND OR NOT	yes	some	none
TDMS	AND OR NOT	yes	some	A & E

Legend: A - Arithmetic Functions
E - Exponential Function
P - Parentheses

Figure 2.1 Nonprocedural Language Operators.

```
   TRADITIONAL              NONPROCEDURAL
   PROGRAMMING               PROGRAMMING
     PROCESS                   PROCESS

+---------------+         +--------------
|    report     |         |    report     |
|  definition   |         |  definition   |
+-------+-------+         +------+-------
        |                        |
        V                        V
+-------+-------+         +------+-------
|  scheduling   |         |   running     |
|      of       |         |      of       |
|  programmers  |         |   program     |
+-------+-------+         +------+-------
        |                        |
        V                        V
+-------+-------+         +------+-------
|    coding     |         |   results     |
+-------+-------+         +--------------
        |
        V
+--------------
|    program    |
|   debugging   |
+-------+-------
        |
        V
+-------+-------
|    running    |
|      of       |
|    program    |
+-------+-------
        |
        V
+--------------
|    results    |
+--------------
```

Figure 2.2 Programming Comparison.

Application Development Facility (ADF) showed a beginning programmer can be much more productive with ADF than with procedural languages, less initial training is required, and it seemed easier to learn. The data base accessing logic is predefined in ADF, thus providing consistent and generally efficient call patterns. The use of conventional procedural code requires coding of the data base calls in every module, thereby running the risk of coding inefficient call sequences. The programming man days required for several projects indicated an average 12-fold time saving of actual ADF time against estimated COBOL time. This claim does require qualification, however. The man days estimated for COBOL programming actually included two functions--inquiry and update. Both functions are essentially equivalent in ADF; therefore, separate ADF programs did not need to be written in order to provide both inquiry and update capability. [Ref. 8: p. 168]

Data description does not commit us to the internal representation of data within the computer, that is, the user is not required to develop a conceptual view which corresponds to the actual way data are stored. One of the functions of a nonprocedural language is to interface between the two. The NPL also facilitates the adding of data or the reorganizing of it. Systems like FOCUS and RAMIS II are general purpose, in the sense that they are intended for use in a considerable variety of applications. A quite different sort of nonprocedural software development is embodied in the specialized applications packages offered by their vendors. There are a great many of these and probably the list will continue to grow. See Table III. One major problem in the implementation of nonprocedural languages is the "ripple effect." When bugs are found or a change is made in a particular version of the software package, modifications are made difficult by equipment type or brand peculiarities. The new versions have to be adapted to each type of hardware gear.

It would be misleading to suggest that nonprocedural languages are a panacea to the ills of the software world. Although they are widely acclaimed, there are some applications for which they are clearly not suited. However, the wide acceptance and user satisfaction which they enjoy indicate that they are a legitimate and effective addition to the information processing arena. Nonprocedural languages have demonstrated their ability at database

TABLE III

Specialized Nonprocedural Language Packages

1. High resolution graphics
2. Statistics
3. Full Screen Data Entry and Display Applications
4. Procedural Language Interface
5. External File Interface
6. Usage Accounting
7. Communications Interfaces
8. Financial Planning and Modeling
9. Information Management System Interfaces
10. APL Interface
11. Formatted Screen Manager
12. Interactive Request Modification
13. Word Processing

management, report generation facilities, handling of ad hoc queries, interfacing with host languages, and handling of various file structures. Even so, there is more to be considered here than ease of use and powerful capabilities. An important consideration is the ability to adapt to change. How flexible are these nonprocedural languages? With COBOL, a relatively minor change in programming logic or report formatting can be a headache of several days work, not to mention the tediousness of the job, its scheduling, and the ever-present possibility of making errors. With nonprocedurals, total reorganization of internal storage can be accomplished relatively simply. But the real beauty lies in the fact that after a total reorganization, changes in data operating commands are not necessary.

Continued exclusive use of procedure-oriented languages results in low programmer productivity which just cannot keep pace with the demand for new applications. The driving force behind these new programming methods is the cost and difficulty of traditional programming methods. For the time being at least, nonprocedural languages seem to be providing long needed relief by maximizing integration of user services in a user friendly manner. Implementation of a data dictionary is a time and headache saving administrative strategy. A data dictionary is a file stored in the data base, and accessible by the various users in an interactive manner. It provides a narrative record which describes the name, aliases, uses, format, access authorities, and so on, of the data item. It is a major step forward to integrate development tools by extending the standard language, coupling it to a data dictionary and DBMS and supporting the development process in an interactive environment. These systems offer the potential of significant productivity gains through ease of use, the convenience of all development services at a terminal workstation, the completeness of modern language and the leverage provided by integrated facilities. [Ref. 6: p. 42] When organizations properly utilize these new tools, it is expected that significant productivity benefits will result.

III. HUMAN FACTORS IN LANGUAGES

A. INTRODUCTION

Computers today are providing an expanding range of services to a rapidly growing pool of users. Electronic mail, document production, and information retrieval are widely used services. Such facilities make our lives easier and can enhance the output of many users. Yet a bottleneck remains which hinders the wider availability of such systems and decreases the effectiveness of those presently in use; this bottleneck is the man-machine communication barrier. Simply put, a major complaint against today's systems is that they are not very good at communicating with their users. They often fail to "understand" what their users want them to do and then are unable to explain the nature of the misunderstanding to the user. In fact, it is the common experience of users of interactive systems, whether novice or experienced, infrequent or regular, that communicating with their machines is a time-consuming and frustrating experience [Ref. 9: p. 19]. Various levels of performance can be achieved, given various degrees of hardware capability and programming ingenuity. In the short run, the issue becomes one of performance/cost tradeoff, influenced by the requirements of the application. In the long run, declining hardware costs and more skillful programming will provide better performance for less cost [Ref. 10: p. 14].

B. DESIGNING FOR THE USER

Organizations run on information. Information is more than the mere summation of collected data; it is a complex structure of interdependencies and relationships, which need to be presented in an understandable format constrained by contextual, accuracy and timing requirements. Late, inaccurate, incomplete information is of questionable value. The thrust of management's attention revolves around reliable information and effective ways of obtaining it. The key is people productivity. Efficiency in speed, cost and reliability have traditionally been yardsticks associated with measuring machines. However, machine efficiency is of little value if it cannot be properly utilized because of inefficient users. Therein lies the reason that management is focusing its attention on developing efficient and productive users. The rapid growth of the computer field has caused computers to become cheaper and more available; interactive computing is in use in many businesses, and home computing is becoming more and more commonplace. As a result of these developments, a new breed of users is emerging--the nonprogramming computer user. In order to facilitate this type of "programming," a mechanism must enable a human being to express algortihms naturally and succinctly as well as clearly and completely. With several hundred programming languages having been developed over the past 30 years, clearly what is natural and succinct to one person may not be so to another [Ref. 11: p. 53]. It is often assumed that, ideally, computers should be programmed in natural language. Schneiderman [Ref. 12: p. 206], believes that the use of computers would be facilitated if natural language systems were available. Users would not have to invest in learning programming or database query languages and struggle in translating their thoughts into an artificial language. This is possible at the present time, although processing is very costly, and computer time is inefficiently used. This postulate of natural being better has been refuted by Small and Weldon who studied English vs. SQL, concluding:

> The common assumption that ordinary, everyday English is the ideal way to communicate with computers is not supported by present results. Subjects were not reliably more accurate using English than using SQL, suggesting that the structured language is easier to use [Ref. 13: p. 61].

Studies indicate that a formal nonprocedural language helps structure user requests. English may be too flexible, inappropriate for queries, or perhaps a natural language is not a natural query language, as Montgomery suggests. [Ref. 14: p. 1075] This would be attested to by anyone familiar with legal documents and the painstaking detail which must be employed in order to present a precise meaning. Reading and comprehending a natural programming language is relatively easy, but writing syntactically correct code is a challenge. The closeness of natural language to English makes it difficult to remember the grammar of the natural language, an example of proactive interference, the confusion between what you know and what you are trying to learn. The closer the two resemble each other, the greater the proactive interference [Ref. 12: p. 199].

Watson [Ref. 15: p. 1], characterizes English as a difficult language to use to describe things with precision, and; therefore, a poor choice for delineating computer specifications. While that point is arguable, it is hardly worth debating the merits of developing nonprocedural lnaguages in non-English vocabularies, foreign or synthetic. Easy learnability would be eliminated, and user resistance is bound to be high.

As mentioned earlier, vendors of nonprocedural languages propose that use of their products will relieve programmers from redundant and repetitive activities, thereby offering them a chance for greater programming productivity as well as offering the nonprogrammers a chance to get into the world of automated data processing and improve their productivity. The use of programming productivity improvements can reduce systems development cost by as much as 50% and program maintenance cost by as much as 75% [Ref. 16: p. 28]. Their increasingly widespread use throughout a wide variety of industries would suggest that there is some credibility to these claims, or at least that this is the type of tool for which industry is looking. But what is it that constitutes a successful nonprocedural language? They are not, afterall, carbon copies of one another. Some aspects must be indicative of better success than others. Mere implementation of a nonprocedural language is certainly no assurance of its success. If a gap exists between what the user expects and what the system delivers, the system could be judged a failure despite the technological soundness of the system [Ref. 17: p. 42]. Design based on user needs is a non-trivial concern.

Moynihan [Ref. 18: p. 116], states that success can be measured in two ways: first, in the case where a user can choose whether or not to use a system for a particular job, success would be measured by extent of use, and second, in the case where the user is obliged to use the system, success is measured by the user's overall degree of satisfaction. The latter would be applicable to the nonprogrammers, since their only access to the data base is through the nonprocedural language. We are again faced with an enigmatic evaluation of "satisfaction." How then can this "satisfaction" be measured? Several authors have generated scorecards on the subject. Watson [Ref. 15: p. 4], states that simplicity, little requirement for memorization, freedom of conceptual view, and timeliness are the essential elements. Hopper [Ref. 19: p. 3-4], describes user satisfaction in terms of ease of use, clarity, and portability. Synnott and Gruber [Ref. 20: p. 192], emphasize accuracy, timeliness, ease of use and responsiveness. Hayes, Bell and Reddy [Ref. 9: p. 27], stress the importance of flexibility, help facilities, and personalization in the form of freedom of conceptual view. Reisner [Ref. 7: p. 13-31.], has devoted an entire paper to the study and evaluation of ease of use. These schools of thought are overlapping, but none seems to be all encompassing. Moynihan [Ref. 18: pp. 116-118], through empirical studies, has composed a comprehensive list of eleven key points to be followed in order to ensure that the system is designed with the user in mind, an inherent trait for a successful system. Tapscot [Ref. 21: p. 132], concurs that success is a function of user-driven design methodology.

The key points are:

1. The system should be forgiving when the user makes mistakes.

2. The system should be dependable.

3. Users should have easy access to the system.

4. Users should get any help they need to use the system well.

5. The system should not damage users' jobs or make users feel unimportant.

6. The system should not make users feel isolated.

7. The system should not make users feel overexposed to scrutiny.

8. The system should not make it hard for users to escape from their jobs.

9. The system should not create unfinished business for users.

10. The system should behave like a machine, not a person.

11. The system must be important to the user.

Each of these points shall be addressed in turn.

1. The system should be forgiving when the user makes mistakes. The system needs to offer the user helpful instructions to recover from any errors he may make. Non-procedural languages offer prompts and help facilities which provide explanations and elaborations on correct format, acceptable field entries, allowable words, etc. at several levels.

2. The system should be dependable. System errors cause users to lose confidence. On the other hand, a system which is flexible and amenable to change can be a joy to use. Potential use of the language in new and unforseen areas must be considered. It should be viewed from the point of possible extensions to meet other needs. Users' views on its applicability in actual practice, the efficiency of the implementation, its potential for expansion into other, and probably unforseen, application areas, ease of training and effectiveness of documentation, and problems of conversion and compatibility all play key roles.

3. Users should have easy access to the system. The system needs to be easy to learn, and, additionally, quick and simple sign-on procedures and rapid response times are essential. Being oriented toward particular types of applications, nonprocedural languages generally require less time to learn for efficient usage, tend to minimize or eliminate specification of computer and interface operations, permit

concentration on the ultimate process control strategy and are self-documenting [Ref. 2: p. 124]. The introduction of nonprocedural languages hopes to make nonprogrammers a new pool of automated information processing personnel. Since such a user lacks computer experience, a successful query language should be easy for him to learn, use and remember.

4. Users should get any help they need to use the system well. This includes sufficient training, in house experts, and up to date manuals. Fill in the blank or menu selection facilities make computer use possible without any training at all. On-the-job training is very important for programming trainees; they need close guidance. Users also need to feel that there is someone they can turn to for help. One product of a user satisaction survey revealed that divisions within an organization which had internal people knowledgeable about the system had a higher level of satisfaction with the system than those divisions without any staff professionals [Ref. 20: p. 192].

5. The system should not damage users' jobs or make users feel unimportant. The system must not supplant human judgement. Nonprocedural languages decide on how to accomplish the task it has been assigned, but it is the human who decides upon and assigns the task.

6. The system should not make the user feel isolated. Users will certainly have a bad attitude towards the system if they consider themselves to be involuntarily glued to their terminals to the exclusion of any human interaction.

7. The system should not make users feel overexposed to scrutiny. This point deals with management attitude towards lower echelon workers, not with any actual trait of the language. However, workers can be expected to be disgruntled with the system if they find their bosses constantly monitoring them.

8. The system should not make it hard for users to escape from their jobs. This point is mainly aimed at managers, and is concerned with the portability of terminals which allow the managers to do work away from their offices.

9. The system should not create unfinished business for users. When a job has been programmed, the user needs to feel a sense of finality; the need to make additional adjustments and postings is a thorn in the side which is eliminated with the use of nonprocedural languages.

10. The system should behave like a machine, not a person. This is not meant to discount the value of user-friendliness and understandable "dialogue" with the computer. Many users find terminals which are too talkative offensive or unnerving. Building computers that behave like people is like trying to build planes that flap their wings.

11. The system must be important to the user. This point may well be the most influential of all, since all the others build to it. If any of the others fail, the users may tend to disregard the capabilities of a very beneficial system. Users will only seriously consider the system if they feel that it will help them to do a good job, and if they are not put off by lack of understanding how to use the system. Ease of use is of tremendous importance. A carefully designed user acceptance methodology can successfully minimize the gap between the system a user expects and the one which is delivered, resulting in significant improvements in productivity [Ref. 17: p. 44]. All the bells and whistles the computer designers can create are of dubious value if users cannot or will not use them. The systems must overcome resistance to change, increase understanding, and convince users that it is for their good.

C. THE EASE-OF-USE ISSUE

Since the nonprogrammer generally will lack computer experience and possibly use the language as only a portion of their jobs, somewhat intermittently, a successful nonprocedural language should be easy for him to learn, use and remember. Data base access is significantly eased if the user does not need to deal with the data base in terms of unfamiliar structures, but can think instead of it in his own terms. A nonprocedural access language enables a data-base user to identify and select those items in the data base with which he is concerned by stating properties they are to possess, rather than by specifying how they are to located. The significance of ease of use is documented in the numerous studies done in this area.

Human factors methodology has been applied to computer equipment, but it has been focused largely on physical devices (keyboards and display design) rather than on cognitive factors, which are more appropriate to measuring ease of use of nonprocedural languages. A major problem in extending human factors methodology has been to develop a definition for the ease-of-use of a nonprocedural language that corresponds to intuitive notions of ease-of-use and permits measurement in a feasible amount of time, with some approximation to scientific rigor. Further, nonprocedural languages are complex and involve cognitive activities (learning, understanding, remembering) rather than only physical and perceptual ones. [Ref. 7: p. 16] This is a tall order, however, and has yet to be filled.

Nonprocedural languages differ in ways that may affect their ease of use; namely, they are:

1. Syntactic Form--With two-dimensional form, users write queries by filling in forms on CRT screens. Linear

syntax is written in normal left to right, top to bottom fashion. Two variants of this syntax are shown in Table IV. SQUARE employs a positional linear syntax while SQL uses a keyword.

2. Procedurality--Experiments of Welty and Stemple using TABLET and SQL conclude that people more often write difficult queries correctly using a procedural query language than they do using a nonprocedural query language, that is, specifying a step by step method for achieving a result as opposed to describing a desired result without specifying how it is to be achieved. The experiments found no statistically significant difference in the ability to write easy queries. However, there is a statistically significant difference when difficult queries are used. Their results show a clear difference in the ability of students having little or no experience with computers to learn the two different languages. They conclude that the cause of the difference to be the concrete procedural model underlying the TABLET queries and missing in the less procedural SQL queries. They believe that the TABLET users were encouraged by TABLET's procedurality to think in terms of concrete procedures that change tables of information, and that this allowed them to perform somewhat better. Other results also clearly indicated that exposure to languages designed for expression of procedures, BASIC and FORTRAN, gave students experience which helped them retain TABLET but not SQL. Additionally, a second experiment showed that the more procedural language was easier to learn for students with no previous computer language exposure [Ref. 22: p. 640].

3. Data Model--The relational model imposes little constraint on the way that a user is able to interpret and utilize data. There are no complex tree or network

TABLE IV

Examples of Nonprocedural Language Queries

Query Language	Example Query: Find the names of all employees in department 50 who earn more than $50,000.

SQL	SELECT NAME
	FROM EMP
	WHERE DEPTNO = 50
	AND SALARY > 50000

QBE	EMP NAME DEPTNO SALARY print 50 >50000
SQUARE	$_{\text{NAME}}\text{EMP}_{\text{DEPTNO,SALARY}}$('50',>'50000')
IQF	(1) FROM EMP FILE (2) FOR DEPTNO 50 (3) AND FOR SALARY> 50000 (4) LIST NAME

FOCUS	TABLEFILE EMP
	PRINT NAME
	IF DEPTNO IS 50
	IF SALARY GT 50000
	END

RAMIS II	TABLE
	FILE EMP
	PRINT NAME
	IF DEPTNO IS 50
	IF SALARY GT 50000
	END

TABLET	FORM DEPTFIFTYRICH
	  FROM DEPTNO OF EMP
	  AND SALARY OF EMP
	KEEP ROWS WHERE DEPTNO IS 50
	  AND SALARY GT 50000
	PRINT NAME

structures that coerce all users into a particular limited view of the relationship, i.e. a single logical view of the data base. A virtue of the relational model is its simplicity and ease of description for all users. A general purpose data management system should allow the user the expressive power of the network model and allow naive users to pretend the data base is a collection of relations [Ref. 23: p. 808].

D. THE ISSUE OF PRODUCTIVITY

Throughout industry, the most widely chosen method of solving the problems of programmer productivity, responsiveness to end-user needs, increasing information demands, and changing data requirements, is to add more people. As a result, system development has become one of the most labor intensive processes in American business. The fallacy of that approach is proven by Brooks [Ref. 24: p. 13-26], by showing that adding manpower increases communication needs and actually results in degradation, at least initially, of productivity. There is a natural tendency to equate hard work with productivity, but this is not the case; personnel effort does not equal productivity. The key is not necessarily to work harder but to work smarter, more efficiently [Ref. 25: p. 21]. Productivity is a function of people, even if it is accomplished through a machine.

Welty and Stemple [Ref. 22: p. 626], agree that humans remain the crucial part of the system. Efficiency in the use of a system can be ineffective if the system is not designed to match the needs and abilities of its users. This fact has led them to explore research involving the human oriented aspects of computer languages previously cited. Software that will enable a manager to enter input and generate output from a terminal in a conversational mode and in a language close to English is the trend of the future. Business people want simple, versatile programs that are reliable, readable, verifiable and maintainable [Ref. 13: p. 70].

The data relevant to various human factors design problems cover a wide span and exist in many forms, such as the following:

- Common sense and experience, such as the designer has in his "storage," some of which may be valid, and some not,
- Comparative quantitative data, such as relative accuracy in reading two types of visual instruments,
- Sets of quantitative data, such as measures of samples of people and error rates in performing various tasks,
- Principles, based on substantial experience and research, that provide guidelines for design,
- Mathematical functions and equations that describe certain basic relationships with human performance, such as certain types of simulation models,
- Judgement of experts,
- Design criteria, consisting of a checklist of specifications [Ref. 26: p. 458].

Numerous experiments in this area concerning nonprocedural languages are based on quantitative data, although industry in actual practice has been found to rely on the judgement of experts.

From numerous studies and experiments, a laundry list of essential features a computer language should have if it is to interact gracefully with its users has been established. It is not desirable that an ideal language should mimic a human style of communication. Rather, the system should satisfy the communication needs of its users in the way that makes best use of the available technology. [Ref. 9: p. 29] No industry wide performance standards have been established for traditional programming productivity measurement, let alone for the newer nonprocedural languages. Typical examples of such measures would include output per unit of time, variance in scheduled vs. actual time and resources to completion, degree of fulfillment of requirements, and minimal necessary maintenance. For right now, all we have to go on is the judgement of managers, who, though lacking any hard and fast statistical data, have decided that nonprocedural languages are a great boon to productivity.

E. TO USE OR NOT TO USE

These languages can offer significant reductions in complexity and cost for both the applications programmer and the non-expert in accessing the database, but they are not the be-all and end-all of programming languages. One must bear in mind that many of the "new" features of the nonprocedural languages have been available for years. Some early languages such as SPECOL were designed specifically for file manipulation. Mark IV and INQUIRE are designed as file management systems; Mark IV is an effective report generator.

The rules specifications of ADF seem to be learned quite readily, although a new vocabulary is introduced, different naming conventions are used, and various rules must be tied together to form an executable ADF system. New programmers tend to be confused until they have written and implemented an application themselves. However, the use of ADF has been rejected by one company for several projects because of deficiencies in standard processing that would have required special processing to overcome. The inability to access multiple data bases and multiple hierarchical paths and the access to only one occurence of a segment type are two key restrictions that have limited applications. [Ref. 8: p. 172]

Most nonprocedural language vendors have a myriad of tales concerning queries that require a handful of instructions, as opposed to the hundreds or thousands of lines of code which would be required if programmed in procedural language. This is possible because the language is able to make a lot of assumptions about the task at hand. A more complex query for such a language is one where some or many of the assumptions are untrue and therefore exceptions. Under these circumstances, it is questionable as to whether the simplest solution involves using the English-like language. In fact, it may be faster and easier to write the program in COBOL if the query is complex. [Ref. 15: p. 1] Despite the claims made by their suppliers that many nonprocedural languages are simple and "English-like," there is good reason to doubt whether a command mode is a suitable interface for more than a proportion of the possible users of a computer information system. Quite a few of today's nonprocedural languages have recognized this and provide a dialogue (question and answer) mode. [Ref. 4: p. 15]

Computers have impressive speed, storage and accuracy which are bypassed if we use natural language. User knowledge of the application domain seems to be critical; without this prerequisite, natural language usage would be extremely difficult.

F. SUMMARY

As the number of potential computer users increases, it becomes necessary to reexamine programming language concepts. Rather than develop more powerful procedural languages, descriptive, very-high-level languages may be a better approach. Very-high-level languages allow the novice user to describe the problem rather than the method for its solution. The user is not rquired to transform a conceptual data model, which is problem oriented, to a machine oriented model. [Ref. 12: p. 116] Many feel that user software enhancements represent the wave of the future and that there will be little, if any, need for practitioners. Considering the growth of computer sites, as well as the growth of applications at existing sites, this prognosis is hardly justified in its entirity. What is reasonable to expect is that the data processing environment and the skills utilized by data processing practitioners will change dramatically. No matter how powerful computers and software become, it is hard to imagine eliminating the need for a professional staff to direct and improve the data processing function. It is logical to expect that the nature of the practitioner's position will shift from a programming orientation to an analytical one. It is arguable that software quality would improve greatly if greater emphasis were placed on analysis rather than upon programming today. [Ref. 15: p. 4]

If shared management of information resources will be the trend in the 1980's, we will need some strategies to get information management involved with users and users involved with information management. This is not meant to discount the viability of traditional procedural programming, the value of third generation languages, or the employment perspective for computer programmers.

Non-procedural languages and the new pool of "programmers" they produce are a complement to traditional programming, not a substitute. There are some applications for which the nonprocedural approach is clearly not applicable. Additionally, there generally is a point in the complexity of any class of problems beyond which procedural specification is simpler than nonprocedural [Ref. 22: p. 628]. It is clear that computer usage cannot feasibly be limited to relatively few specialists. Equally, it is not viable to require a high level of computer skill for the performance of all computer processing. Now that the computer is no longer viewed as a stellar wonder, accessible only to data processing experts, industry has come to realize that the computer is a tool, albeit a sophisticated one, and that naturalness for the user in the development of programming languages is an issue which must be addressed.

IV. ANALYSIS OF RAMIS II

A. INTRODUCTION

Chapter V and this chapter are essentially the backbone of this thesis and the basis for the choice of the particular topic. This chapter will deal with the industrial response to RAMIS II as Chapter V will deal with FOCUS. From this industrial response by the user organizations of RAMIS II and the subsequent analysis of that response, certain questions will hopefully be answered and offer enlightenment as to whether the purposes for acquiring such systems have been fulfilled. Through questionnaire submission and reply and/or personal interviews with the user organizations listed in Appendix A, determination will be made as to the value of RAMIS II and its related system/components to be discussed later.

The structure of this chapter will first deal with the vendor company and the products and services it offers and/or attempts to provide, followed by an examination of the industrial response and a conclusive determination as to user satisfaction with the vendor, the product, and the resultant productivity gains within their respective organizations.

B. MATHEMATICA PRODUCTS GROUP (MPG) AND RAMIS II

Although RAMIS II, RAPID ACCESS MANAGEMENT INFORMATION SYSTEM, was first developed in 1967, Mathematica Inc. did not form Mathematica Products Group (MPG) until 1975. It was formed solely for the purpose of developing, marketing, distributing, and servicing the RAMIS II product/s. The actual RAMIS II nonprocedural product we know today was released to the marketplace in 1977. According to the marketing packages obtained and from interviews with branch management, Mathematica does more than simply produce software. It also provides consultation services, and conducts policy research in conjunction with the software production process. Through what they term "ongoing research" in the areas of business management and computer technology, MPG attempts to provide a comprehensive , easy-to-use management information system which also provides effective and efficient use of related computer resources.

Their belief is that the increasing demand on programmers' time has created the need to simplify the programming process to allow for increased programmer productivity and increased nonprogrammer use of installed systems. Rapid advances in computer technology have also created the need for adaptability to constantly changing programming environments. These demands by today's industries have led to the massive efforts in the creation of marketing enhancements by MPG and the other nonprocedural product manufacturers.

MPG is involved in a continuing process to develop, expand, and refine their RAMIS II product lines. Their marketing approach stresses to present and prospective clients their practice of using seventy (70%) percent of their research and programming effort to improving their present product and adapting it to future generations of computing equipment. The other thirty (30%) percent of the research effort is dedicated to analysis of the future requirements of business and the subsequent MPG product release to meet those requirements.

In line with this apparent dedication to continually improve their product line is their product usage restriction to IBM related equipment. So long as their clientele maintains their usage of IBM compatible equipments, MPG will be able to continue to provide services. This aspect of MPG is aided by maintaining a close contractual and on-test-site relationship with IBM for their prereleased software and hardware products. This relationship is mandatory in that MPG must also meet contractual relationships with its users.

The product that MPG produces is the RAMIS II system and its related nonprocedural language, a combination of a Data Base Management System (DBMS) and a 4TH Generation Language to (1) allow for a simplification of communications between the user and the computer using English like queries where the user states "what he wants" vice how to do it; (2) allow ease of access to information by all users; and (3) also allow for application generation enhancement to simplify the programming process and consequently increase application programming productivity. In their own words RAMIS II "....combines a comprehensive data base management system with a proven, easy to use, nonprocedural computer language for report preparation and records maintenance."

RAMIS II promises to provide the user with the ability to examine more information and combinations of information to form more valid and appropriate decisions; to access information more quickly; to create new systems and applications in approximately one-fifth the time of procedural languages; and to make queries and receive replies from the database more rapidly.

In conjunction with these abilities, RAMIS II promises the user the benefits of less environmental maintenance of the database; data independence to reduce maintenance and user constraints; increased storage and computer rescurce efficiency; faster implementation; cost savings; portability; and the bottom line productivity enhancement.

To accomplish the aforementioned items, RAMIS II offers an integrated package of features from which the user/consumer can construct a system that will be designed to meet his exact requirements. The features of the basic package and optional components and their related descriptions are listed below. The costs/prices of these packages/components are given in Appendix D.

1. The Basic RAMIS II System

a. RAMIS II Data Bases--provide for the storage of data for present or future application use.

b. Database Manager--allows for efficient data access and use as well as effective use of human and computer resources through the use of the RAMIS II nonprocedural language.

c. Nonprocedural Language Processor--allows for application generation, report preparation, and file maintenance by use of the NPL.

d. Interactive Request Modification--allows for corrections of erroneous requests and generation of requests for additional data by revision of previous requests.

e. Screen Manager--allows for control and manipulation of the screen environment.

f. Operating System Interface--allows for operation of RAMIS II and the ability to run under all operating systems that are IBM mainframe compatible.

2. Optional Components for RAMIS II

a. Procedural Language Interface--allows the use of procedural languages to manipulate and maintain database records.

b. GRAPH and High Resolution Graphics--allows for production of charts and graphics on output devices.

c. External File Interface--provides for access and reporting on files external to the RAMIS II system by declaration in the RAMIS II File Dictionary.

d. Automatic Interfaces--allows for access to information stored on a variety of other data bases including ADABAS, DL/1, IDMS, IMS, and TOTAL. Can retrieve information using RAMIS II for further processing.

e. READ OS/DOS--allows users from the VM/CMS environment to access from CMS to other data bases maintained by other operating systems running under VM.

f. Financial Planning--allows analysis of information under various means to obtain complete objectivity studies and the production of various financial reports.

g. APL Interface--allows a user from the APL environment complete RAMIS II capabilities.

h. Executive--allows cataloguing of RAMIS II requests for further processing at a later date. Can be used to create dialogues and menus for processing purposes.

i. Usage Accounting--is a tool for the Data Base manager and administrator to fine tune and administrate the DBMS based on monitoring, documentation, and analysis of application use.

j. Interactive Request Modification/Extended (IRM/E)--"extends" the IRM capability by allowing development, storage, and cataloguing of requests.

k. Communications Interfaces--allows for operation of RAMIS II in a timesharing or teleprocessing environment with RAMIS II or a procedural language.

l. Integrated Communications Interface--"provides an effective alternative to CICS and ICCF. As a complete RAMIS II terminal network manager, it is both more efficient and easier to use."

3. Future RAMIS II Products/Components

a. RELATE--will be a relational model to allow combination of data from various sources and provide a "practical data manipulation tool."

b. Formatted Screen Manager--will allow the user easier construction of user application menus for data entry, change, and processing.

4. Training and Services

Besides providing the product and services previously mentioned, MPG offers extensive training and post sale and implementation services. Training offered ranges from the basic beginner's classes to the more advanced processing classes using RAMIS II. They offer individual and group training at single and package rates. They conduct classes in-class or on-site. Examples of training and classes offered are given in Appendix C. The cost of this training is given in Appendix B. Their post-sale services include on-site counselling and assistance, a consumer HOT-LINE to handle serious and urgent problems rapidly, and direct access to/assistance from the numerous branch offices where they promise to solve your problem expeditiously or have a representative on-site within 24 hours to answer questions and/or correct problems in the installed system.

MPG's marketing packages, pamphlets, seminars, and their trained knowledgable personnel present a pretty picture of RAMIS II, as can be seen by these introductory statements, of which many were taken from actual RAMIS II marketing abstracts. The tools they work with are up-to-date, well documented, and from acceptable/reliable sources. With their marketing aids and tools an inexperienced salesman would have an excellent chance of selling their product; however, the personnel working for MPG are knowledgeable and well trained, and present a good product which they truly appear to believe in, in an excellent manner. Their companies sales record is demonstrative of this ($50 Million in cumulative sales in 1981) as is a quotation, albeit biased, from MPG's president Richard H. Cobb.

No other systems software has ever achieved this level of sales and recognition in so short a period of time.

The next section presents the response by users of the RAMIS II product to determine if the promises made by MPG and RAMIS II have been kept and the obligations fulfilled.

C. ANALYSIS OF INDUSTRIAL RESPONSE TO RAMIS II

The following analysis is divided into the same sections as the questionnaire in Appendix B used to conduct the research. Using this format will hopefully aid in making the comparison and analysis of the industrial responses clearer and more meaningful.

1. Implementation

The majority of the companies interviewed had fairly recently acquired the RAMIS II system. The size of these organizations ranged from 350 to 10,000 people with the number of actual users in each organization ranging from 100 to 2500. There was no relationship between the total number of people in the organizations and the total number of users; the number of users, types of users, and other DP need determinations were a direct result of the quantity and the complexity of the organizations' business transactions. The type of work the company did and the particular job descriptions of employees generally dictated the need for computer usage. The amount of data processing and computer usage by these organizations was usually quite large and an integral part of their business; however, in all but one of the companies interviewed, the amount of data processing done using RAMIS II was between 5 and 10 percent. It is worth mentioning that the company that did not fall in this category conducted 50% of their DP using RAMIS II and had an entire system dedicated to the task.

The way that RAMIS II was chosen over other competitors is also of interest. In most cases, the detailed cost and product analysis done prior to acquisition, was done by three or four managers and/or data processing representatives sitting down over a pot of coffee saying to each other--"I like this."-- "I don't like this."--"This would be nice."--"Let's get this one" without any formal quantitative analysis.Some companies actually did conduct cost/benefit analyses and compared the various products, the vendor services provided, system capability, and other relevant aspects; however, even in these organizations, the final decision was made by those three or four managers sitting over a table and a pot of coffee hashing it out.

Depending on the their needs, the various competing products mentioned by the companies interviewed included other nonprocedural languages, report generating products, query languages, and other database systems like FOCUS, NOMAD, SYSTEM 2000, ORACLE, INFO, SQL, ADRS II, EASYTRIEVE, AND MARK IV. The reasons for the choice of RAMIS II included availability, user friendliness, RAMIS II"s compatibility with their system, RAMIS II's product mix, the type of processing required, vendor services provided during and after sale, and in two cases "vendor interest and involvement. Since many of these systems are very recent developments, and at the time of acquisition the choices available were few; but in almost all cases, the companies said that if they were presented with the same decision currently, the choice would still be the same.

The work accomplished using RAMIS II varied slightly among the organizations, but the majority used the system for data analysis, management decision support, report generation, and ad-hoc inquiry. The more technically oriented firms have constructed detailed models and prototypes using the system. The users of the systems ranged from clerical staffs to middle managers. In only one of the companies was there any use by top management. Generally high level managers submitted requests to the DP department or instructed their staff/s to obtain the desired information. The majority of the users tended to be the staff personnel who processed and retrieved the desired information for management. Also the number of personnel and manpower levels in these organizations was not reduced due to system installation; RAMIS II simply provided another tool to work with. However, to prevent misinterpretation, manpower levels could have been reduced because production and processing capability of existing users was increased in most cases almost five-fold using what one interviewee termed "a conservative estimate."

Contrary to the vending company's beliefs, the system cannot simply be loaded into the computer and be ready for use. The amount of time to get the system into "full swing" varied. Some companies achieved desired results within three months of installation while others required over nine months and were still unsatisfied with the results which one termed "failure to meet their own foolish expectations." Also it appeared that the level of difficulty of applications desired was a determining factor in the organizations' attempts to have the system up and fully supported. The higher the level of difficulty of the target applications, the longer the implementation time. This was generally not pointed out to customers, but they are told that to achieve certain levels of expertise requires learning the system well through usage and training which is the subject of the next section.

2. Training

According to the user organizations, training provided by the vendor was adequate. Training consisted mostly of beginner introductory packages in the areas of "Basic Reporting" and "Basic File Design and Records Management," but courses up to the most advanced skill requirements were offered by the vendor. With a choice of either blanket or per-person rates, most organizations chose to train all users with the introductory package while sending only their counselors and data processing community to the more advanced courses. A few companies determined that using this method would allow difficulties encountered by the resident users to be solved by the counselors and/or the companies' DP people and thus save additional training costs. Other companies set up or are setting up in-house training programs to increase user knowledge and ability without incurring additional training costs; however, the companies that have relied on RAMIS II (MPG) for training and vendor support had the shortest implementation times and have experienced a greater degree of user satisfaction and productivity. By initially choosing vendor provided training, these companies avoided unnecessary delays in both time and efficiency due to excessive user error. These same companies are happy with the vendor training and plan on continuing with follow-up and refresher training despite the additional cost.

During the training and start-up phase for the RAMIS II system, most organizations were faced with additional work backlog, but all but one company blamed this backlog on reasons other than RAMIS II installation. The responses attributed the backlog to poor management and mismanagement of existing resources. Only one company did not have any additional backlog occur. The one organization that did attribute the additional backlog on RAMIS II also happened to be a company that decided to conduct its own training and experienced a longer implementation time.

Initial and refresher training provided by the vendor seems to have been the preferred alternative and the wiser choice as was evidenced by these organizations' greater degree of satisfaction and shorter implementation times. The companies that decided to conduct their own training sacrificed user satisfaction and efficiency for the cost of training.

3. Learnability

The learnability of RAMIS II was again dependent upon what was earlier termed the difficulty of the applications to be produced. The users found the simpler functions easy to understand and execute while the more demanding frequently met with error and required the assistance of others to correct. Although the majority of the users found high retainability, they stated it was due to continued usage, not to the simplicity of syntax of RAMIS II which they also added could use improvement. The users did encounter many problems learning to use the system, but for the vast majority of users, the errors encountered were minor and were corrected by use of manuals or organizational assistance. The areas of greater difficulty often required direct contact with MPG representatives to correct problems encountered and occasionally required the use of the vendor HOT-LINE.

The area of learnability is an area of debate. Some users were enthralled with the languages simplicity while others mentioned room for improvement. There was also no agreement on whether it was easier for a person with or without prior programming experience to learn; however, there was general agreement on the fact that RAMIS II was decidely easier to learn than any procedural languages to the point that a user could write RAMIS II programs after weeks of training that would require months of training with its procedural counterpart.

4. Documentation

The vendor documentation provided to the user organizations was deemed complete and accurate in all instances and rated excellent by three organizations; however, despite the agreement on completeness, there was also concurrence on the lack of structure in them. Although all necessary aspects were covered in complete detail, the majority of users could not navigate through the text to find desired answers. In one organization in particular, a training class was set up for the purpose of teaching the users how to find needed information in the manuals. This deficiency has been pointed out to the vendor in at least three instances and, according to the organizations, has not recieved serious attention thus far. Despite the lack of structure, the general opinion found the documentation quite adequate and user referral to the documents were both commonplace and essential.

5. <u>Vendor Support</u>

The vendor support questions recieved varied responses. In the area of initial considerations for acquisition, many companies listed continued vendor support and services as one of their key reasons for choosing MPG and RAMIS II. In the area of initial package delivery, all but one company was satisfied, and that particular company complained that it was still experiencing problems which had not received their required share of vendor attention. After the system had been installed and "broken-in," most users said that they had few complaints, and the RAMIS II system met their reasonable expectations. Generally they found it to be error free, but periodically found a "bug" which required correction. This periodic problem was the major point of dissension toward RAMIS II services, The problem being that the user's complaint was always solved by what one company termed a "trial balloon," or temporary fix, until a tested solution was released to correct the error possibly 3 to 6 months later. Tied to this was the complaint that these temporary solutions were not distributed quickly enough to the other RAMIS II organizations to prevent similar occurrences. The last item discussed was the vendor HOT-LINE and its usefulness. Most organizations used the HOT-LINE greatly during the implementation and start-up phases, and after they became accustomed to the system, they used it very sparsely. The majority of users professed satisfaction with vendor support, but did not like the "trial balloons" mentioned earlier.

The general opinion found errors to be almost nonexistent, and when they were found, either they were corrected in-house or the vendor provided adequate correction in a timely manner. The previous problem seemed to have been a rare occurrence rather than the norm.

6. Hardware Support/Conversion

In all but one of the organizations interviewed, no hardware support and/or conversion was necessary. All the organizations in the survey already had large systems(IBM 3033's, 3370's, 3081's,and 4300's). The resource requirements for RAMIS II are quite extensive (ranging from 400K to 1024K of memory per user depending upon the overlay structure chosen), but none of the companies interviewed required additional memory or storage requirements to what they already had. According to RAMIS II representatives even the older and smaller IBM systems had no problem in regard to adequate resources.

In some of these organizations after RAMIS II had become an integral part of the existing system, additional memory and/or a different mainframe was procured to increase the efficiency and response time of the system. Also, the computer center managers of the companies knew or determined what combination of resources would best suit operations and took appropriate measures. In one system the RAMIS II usage has evolved and developed to the point where one entire system, the company has two, is dedicated to RAMIS II. A determining factor as to resource requirements lies in the interrelation of RAMIS II to the other parts of the computer installation. In one company the corporate DB is in TOTAL, but is accessed almost entirely using RAMIS II. Every time a report requirement is generated, RAMIS II must create its own DB from the TOTAL system information, and then generate its report. In another organization each user creates and maintains a separate DB for his/her own use.

RAMIS II consumes alot of computer resources, and depending upon the overlay structure, computer system management, and its relation/interaction with other systems, could conceivably consume a much greater amount. Yet by itself RAMIS II has been termed a very efficient product in terms of both response time and resource usage [Ref. 27: p. S11]. The reasons for hardware conversion were almost exclusively inherent to the particular organization, and not primarily due to RAMIS II. Although MPG states its dedication to providing a nonprocedural product that makes the most efficient use of resources, this was not a major concern of the organizations. It truly appeared that with "decreasing hardware costs efficiency is no longer a major concern" [Ref. 28: p. 38].

7. Security/Access

Most of the organizations interviewed do not use RAMIS II security features, but some do use them in conjunction with the operating system's security prcedures/ precautions. RAMIS II offers security with password control, data item access control, and command type control; however, the companies preferred to use their previously installed systems such as RACF (Resource Access Control Facility, which matches each resource to a user set) and various password controls. Some companies used a combination of the operating system's and RAMIS II's security features, while still others only used RAMIS II security with their RAMIS II files. The companies that did use RAMIS II security had no complaints, but they dealt mostly with unclassified data or did not require serious security precautions. MPG claims to provide a high degree of security although the RAMIS II DBMS can be shared among many users.

Control of users that have read/write access and authorization to update/change existing files ensures only authorized changes can be made to files, records, segments, or even fields. Organizations still preferred existing security measures, and the majority of changes made by users were to their own RAMIS II DBs. In almost all cases, with the exception of those having a dedicated RAMIS II DBMS, the corporate DB could not be updated by RAMIS II. Also changes made to the corporate DB were done by the DP departments, not by individual users.

The area of backup procedures and measures were again handled by previously installed measures, in some cases using IMS or CICS transaction logging. Other companies interviewed did not even have serious backup measures installed, but made individual users responsible to ensure that their changes had been made. The companies that did require any type of backup updated the DB usually once a day by batch means not done by RAMIS II. All of the companies operated in such very stabile hardware and software environments that situations requiring such backup occurred about once every six to seven months and had no serious impact on operations.

8. Performance

In the areas of performance and control of the computing environment, RAMIS II offers the beginner to intermediate user a far more productive capability. While constructing simpler programs and making simpler requests to the system, the user has the impression of total control of the computing environment. The more difficult requests and programs tend to alienate the less technically minded users, and these users frequently require technical guidance and assistance from the organization's DP community. This was explained by many of the organizations that the RAMIS II syntax is not logical and has technical ideosyncracies that the more experienced technician would understand, but the beginner would not. One representative explained it in another way by saying that all users are not logically oriented and probably never will be.

As explained earlier RAMIS II offers many packages and additions to their basic package to fit various organizational requirements. A few of the companies interviewed acquired most of the additional packages offered by RAMIS even though they have still used them to date. Also other packages are simply more advanced copies of parts of the basic package adding a bit of redundancy, and the less advantageous copy is never used. Other companies found the basic package so complete that they constructed their own libraries to augment the system in-house. Expected additions to the present package offerings which are eagerly awaited by the users, mentioned previously, are RELATE which will offer users the advantages of a relational DB, and the Formatted Screen Manager which will offer even greater control of the input/output computing environment.

RAMIS II can be operated in both batch and real time environments and, according to the companies questioned, does well in both. Although the majority of the companies acquired RAMIS II to serve in the area of interactive processing for ad-hoc inquiries and report generation, a large amount of batch processing was done also, particularly DB update. All the companies developed and constructed their own application programs to do this processing, but RAMIS will provide that type of service upon request.

All organizations agreed that RAMIS II provided an excellent product requiring little or no debugging. Their estimates indicated 98 to 99 percent reliability, with the 1 to 2 percent error rate being minor in nature and easily corrected. The rare occurrence or exception was mentioned previously under Vendor Support.

9. Improvability/User Satisfaction

This section deals with the organizational view as to whether RAMIS II was an improvement over the previous systems and if they experienced any organizational or user resistance to the new systems.

The opinions surveyed all substantiated a vast improvement over the previous systems; however, in most cases RAMIS II was simply another tool for their existing systems, not a replacement. These companies found RAMIS II to be a great addition, whether it was used to access data from existing databases or to construct their own.

User resistance was almost nonexistent with a few exceptions. One organization covertly introduced RAMIS II under the noses of the DP department, causing great political resistance to the system when the fact was learned. In that particular company there is also a fear that the nonprocedural product will threaten the programmer's job. Another company also experienced resistance to the product, but the resistance came from the nonprogrammers of the organization. It seems that they felt they were being overburdened and being made to do the programmers job also. It seemed to be more of a cultural shock. They were being trained to use the computer and had no desire to be programmers. After the system was installed and working, and the users became accustomed to it, any initial resistance subsided.

10. General

The companies interviewed did not purchase any tailor-made options from RAMIS II although, as previously mentioned, they did acquire most options. The most used parts of the system were again dependent upon the particular organization's uses for RAMIS II and the job descriptions of the users. Most found the NPL processor of greatest value in the basic package. As the users got further away from report generation and more toward modelling and optimization analysis, the procedural interfaces took on greater importance.

The area of RAMIS II that was termed the most error prone was the records management language. It was not that there were errors in the code or the package committed errors, but that the users had the greatest problems in that area. They commented that the inordinate number of user errors in this area is cause for concern and reason enough to influence changes to increase training and improve documentation on the subject.

The last line of questions in this section inquired as to users desires and what recommendations they had for

RAMIS II to improve their product. The majority of recommendations revolved around interactive processing where the terminal prompts the user to determine what has to be done and what the user wants. These recommendations called for the provision of menus to prompt/assist the users in the areas of error correction, application/report generation, and database creation/design(structure,forms,keys). It seems that even with the user friendly/end user product that RAMIS II provides now, the users want something even more friendly.

11. Overall Assessment

The opinion of all but one of the organizations surveyed felt that RAMIS II was probably the best thing that ever happened to their company in the area of data processing. They felt that productivity improvements went beyond the point of increasing the productivity of individual users. They felt that the productivities of the companies themselves were positively influenced by the use of RAMIS II.

These reports of productivity improvements are of particular interest in that many of them vary in reference to measurement criteria. In some companies the productivity growth was gradual while in others the results were more pronounced. One representative noticed that to accomplish the DP requirements for the company no longer required the hiring of additional personnel. Along the same lines another manager was able to decrease his staff requirements by 10%.

Other companies estimated that individual productivity had increased by a factor of 10. This figure was arrived at by observations of programmer output regarding the number of reports/programs which they presently and previously produced. Other organizations observed that programs which normally took 6 weeks to produce using procedural means could now be coded and tested in 4 days using RAMIS II. One organization observed programmers producing systems at 10 times the previous rate. Productivity improvement was also evidenced by the programming backlog reduction within the organizations. All the companies commented that their backlog was being considerably reduced or had been eliminated.

Other performance measures were mentioned also, but were nonquantifiable in nature. These measures included things analysts doing analysis vice programming or clerical functions; increased user availability to information improving decision support; and increased user satisfaction, the implication being that a happy user gets more done.

The vast majority of companies stated that RAMIS II met their expectations and would continue to be an integral part of their organization. Many of the companies surveyed chose RAMIS II as the best nonprocedural product available at the best price. These companies have kept and currently keep track of any new developments in this area and the computer industry in general, and still believe RAMIS II to be the premier product available presently for their users.

V. ANALYSIS OF FOCUS

A. INTRODUCTION

As the workload on data processing departments continues to rise, organizations have taken a second look at the traditional approach to programming in an attempt to survive the deepening backlog of applications for user needs. Productivity increasing tools are the order. The arrival of nonprocedural languages promises to be a software enhancement capable of improving the productivity of both data processing professionals and nonprogrammers through a wide range of design and implementation approaches.

This chapter will deal with the industrial response to FOCUS as Chapter IV dealt with RAMIS II. As in the analysis of RAMIS II, the research for this chapter is based upon the responses of organizations using FOCUS to personal interviews and/or questionaires in the attempt to offer enlightenment as to whether the purposes for acquiring such systems have been fulfilled.

B. INFORMATION BUILDERS, INC. AND FOCUS

Information Builders, Inc. (IBI) was established in 1975 by Gerald Cohen, now president of IBI, and Peter Mittelman, IBI executive vice president, the originators of the first nonprocedural languages. In the short time since then they have grown to a company with seven offices in the United States and with affiliates in Japan, Australia, Brazil, Egypt, and several European locations. A new office is scheduled to be opened in Toronto, Ontario in the near future. A partial list of FOCUS using organizations is listed in Appendix E.

Their product, the nonprocedural language FOCUS, is a high-level, user-friendly, easily understandable, English language information control system which has extensive facilities for complete applications systems development, including comprehensive report generation, file, and data base management capabilities. FOCUS contains facilities for describing both simple and complex interconnected files; for entering, changing, and deleting records in the files; and for preparing reports from information in the files. The purpose of FOCUS is to control an entire application and thereby reduce the need for, or replace, computer programming. The system is structured so that it can be used by non-programmers as well as programmers. FOCUS runs on IBM 370, 4300 and larger, or equivalent mainframes operating under VM/CMS or under MVS with TSO, CICS, or IMS-DC.

IBI's sales revenue has doubled each year, and they presently rank in the top 10% of all software houses. To date in 1982, IBI has sold approximately 420 in-house copies of FOCUS, and an additional 400 copies have been utilized on a time-share basis. IBI expects to reach the 500 copy point for in-house systems by the end of the year, and their marketing personnel project reaching the 1000 point for 1983.

IBI's marketing strategy is still developing, they are realizing that selling a product, no matter how great it is, requires the "promotional push" along with their good intentions. Their printed material, including their promotional literature as well as their user's manual, language primer, and other documentation, is presentable, easy to read and understand, and, of course, paint a glowing picture of FOCUS' abilities. In fact, the FOCUS user's manual is a text sized paperback, a pleasantly portable alternative to the IBM three ring binder approach. However, several other nonprocedural languages whose capabilities do not match those of FOCUS are presented in quite appealing marketing packages, which could feasibly draw the potential customer's attention away from the issues at hand and towards an inferior, or at least less powerful, system. IBI holds regular promotional seminars throughout the country. A general consensus of people attending a San Francisco seminar in October 1982 was that it was too long and too detailed. Instead of presenting general concepts and capabilities, the speaker went into the intracies of actually writing FOCUS queries. After four hours of the same speaker, many in the audience had lost their enthusiasm and interest. Additionally, IBI maintains both a hot-line and the recently introduced help-line to aid in solving customer difficulties with the system. Previously the hot-line answered all customer inquiries. But, they found that the hot-line was being swamped with calls and technical experts were spending time answering lower level inquiries. The help-line now addresses the "run-of the mill" type of questions from users; inquiries of a more demanding nature are referred up to the hot-line for resolution. Now with the help-line taking care of routine problems, the experts can devote their time to more difficult inquiries.

FOCUS offers an extensive list of capabilities and has an impressive list of customers for the short time that they have been active in the market place.

Personnel at IBI are dedicated professionals, albiet prejudiced concerning FOCUS, but truly committed to their product and their customers. They were found to be knowledgeable and helpful and quite eager to answer all questions about FOCUS.

1. The Basic FOCUS System

Basic features of the system are as follows:

a. The FOCUS data base allows files to be structured in a number of ways including hierarchical, multipath hierarchical, network linked, and cross-referenced. Multiple entry point and file inversion are possible at any level. Additionally, any field can be indexed at any level.

b. Online operation with interactive error correction.

c. A user-friendly English-like language for specifying and controlling all facilities and functions.

d. A comprehensive query and reporting capability for ad hoc queries and custom reporting.

e. A Dialogue Manager component to assist in developing prompt-driven interactive procedures.

f. A shared structure data base, supporting both simple and complex multipath and network structures.

g. An easy to use transaction processing language for data base input, maintenance, validation, computation and logging.

h. An interactive data base editor for file browsing and records management.

i. The ability to process QSAM, VSAM, or ISAM files as well as FOCUS data base files.

2. Optional FOCUS Components

a. FOCUS/GRAPHICS for production of high resolution graph forms including histograms, bar charts, point plots, pie charts, and scatter diagrams. It is useable on terminals, color CRT's and flatbed plotters.

b. FOCUS/Statistics for interactive statistical functions, including time-series analysis, regressions, crosstabs, and correlations.

c. FIDEL for full-screen 327x data entry/data display applications.

d. FOCUS/HLI for direct access to FOCUS files from programs written in COBOL, FORTRAN, PL/1, or BAL.

e. FOCUS/FML for production of row-oriented financial documents and reports.

f. Interfaces for processing records from IMS, IDMS, TOTAL, and ADABAS data base structures. The objective of these interfaces is to extend to the non-FOCUS data bases all the facilities of the FOCUS query language in a manner transparent to the user. All FOCUS features which require read-only access to data are supported: printed reports, graphs, statistical analysis, catalogued procedures (the dialogue manager), and the MATCH command, which combines data from several unrelated sources into one report.

3. <u>Future FOCUS Components</u>

a. IBI generates enhancements on an as-developed basis. They generally do not charge their customers for these enhancements which are sent automatically, along with documentation. Major FOCUS releases are produced approximately every 6-9 months.

b. The big news for FOCUS is that it is targetted to be available on the IBM personal computer (PC) by summer 1983. Features will include capabilities for both uploading of PC written applications and data to the mainframe as well as downloading of programs and data from the mainframe to the PC. One shipping industry executive stated that when this capabiltiy becomes a reality, he intends to buy a personal computer, if for no other reason, so that he can still have working copies of data and programs in the event of a mainframe system crash.

4. FOCUS Training

In addition to their basic system and optional add-ons, Information Builders offers training at various levels, ranging from basic beginners classes to advanced processing classes. Individual and group training are available at individual and package rates, and can be conducted at their offices or on-site. On-site training is the preferred since the students will be trained on the equipment with which they will be working. IBI does, however, have ample training space and hardware for meaningful training on their premises. A course listing and fee schedule is provided in Appendix F. In addition to formal training and regular system enhancements, IBI completes their support package with technical expertise, application assistance service, and consultant services. IBI promises that a customer will never encounter a situation where a call for help will not receive full and satisfactory response. In the event of a system difficulty, customers are directed to call and ask for technical help and to expect to receive an immediate response. IBI pledges to stand totally behind its system and service responsibilities, stating that a FOCUS warranty is forever.

C. THE FOCUS USERS GROUP (FUSE)

FOCUS users are enthusiastic, to say the least, about the language, so much so, that they have formed a FOCUS Users Group (FUSE) which holds quarterly regional conferences, a nationwide annual conference, and has a newsletter--all for the purpose of making recommendations for improvements and enhancements back to Information Builders for future FOCUS versions. FOCUS users were more than pleased with the package. All felt that it had at least met their organizations present requirements and anticipated that it would continue to meet future requirments. Most users believe that it exceeded their needs but that the excess capability was sure to be used in the future. No user felt that FOCUS fell short of their organization's present information processing requirements.

D. ANALYSIS OF INDUSTRIAL RESPONSE TO FOCUS

Facets of the user response to FOCUS will be grouped under the same headings used in the previous chapter. Overall ratings of companies using nonprocedural languages are listed in Table VII in Chapter VII.

1. Implementation

The almost unanimous reason for choosing the nonprocedural language approach, and FOCUS in particular, was the ease of use and efficiency of the report generation facilities. Those attending the FOCUS promotional seminar stated that their uppermost concern in digging their way out of their paperwork backlogs was to obtain an efficient and speedy report writer. Other features frequently cited for choosing FOCUS over other languages considered were the database management features and the full screen editor, FIDEL. Most organizations looked at only two or three other languages before actually choosing FOCUS, in the opinion that they had a representative sample and that much more comparison would be time-consuming overkill.

The main use of the system is, as stated earlier, report generation in a wide variety of applications. For example, the City of Fresno, California, uses FOCUS for ad hoc payroll reports, budget preparation, tax computations, business and dog licenses, accounts payable, city clerks' bid and contract reports, building permits and inspection reports, parking citations, paramedic information, and city attorney case histories, to name a few. Generally, FOCUS was found to be used for 70-85% of any given organization's data processing needs. Companies using FOCUS varied widely in size and in both the number and proportion of employees using the language. These differences, of course, had an impact on the speed with which FOCUS was integrated into their operating cycle. Getting into full operating swing with FOCUS was quite variable from one organization to another, dependent on the volume of the backlog and the extent of applications utilizing FOCUS. Generally, The learning curve seems to start quickly and gradually taper off after a few months, followed by new things to learn as improvements to the package and inputs from other users are received. No reductions in manpower have been experienced both because of the backlogs to be cleared out and because of significant increases in applications and reports requested by users in response to FOCUS capabilities. The shortest time span for "full swing" operations was approximately three months; the longest was experienced by a company who has has FOCUS for one year and is not in "full swing" as yet. A company representative stated that their backlog was so overwhelming that they expected to take more than a year.

2. Training

All organizations interviewed opted for the on-site training both because it was more cost effective and time-saving and because their employees would be training in familiar surroundings. Basic report generation and file management instruction which comes with the package as well as the ten day on-site consultant provided a satisfactory start-up in the opinion of most companies. The more advanced training, as would be expected, was chosen only for a select few higher up in the companies' data processing echelon. Most companies plan to have more employees attend IBI basic training courses at 3-6 month intervals. Companies were split about 50-50 on the subject of internal training. While all the companies had at least one employee known as the "resident expert," only about half of them expected to use this person to actually train other employees in the use of FOCUS. The other half had no such plans. However, this well may be a function of personalities or personal preference on the part of the resident expert, who were all of the opinion, as stated by one of them, "There's no way I'm spending my time teaching. It makes more sense to bring IBI here for a couple of days. I'm not going to do it!" During the training and start-up phases, companies experienced slight increases in their backlog but were not concerned over it. The attitude was--"What's another couple of days really going to matter?" They viewed the negligible step backwards as a small price to pay for the substantial steps forward which they expected to experience.

3. Learnability

General concepts of the language were quickly learned. As would be expected, the ease of learning decreased as the difficulty of the application increased. When more statements are required to complete a query, the situation presents more opportunites for mistakes in syntax, formatting, logical thought and sequencing. Actual retention of the language has been found to be reinforced by repetitive use. Frequent referral to the user's manual is experienced in the initial stages. With continued use of the system, manual usage decreases to the point where it is only referred to in the event of a new or infrequently used application. Learnability was found to be satisfactory to those with reasonable expectations, but somewhat disappointing to those with admittedly high expectations. It also appears that those with no programming background do not have a significantly longer or more difficult time learning the basics of the system than their programming counterparts. No data is available on a comparison of more complex applications simply because companies have reserved the higher level capabilities of FOCUS for their more experienced users.

4. Documentation

Overall evaluation of vendor documentation content was excellent. However, the general opinion of the documentation set-up was was fair to poor. The documentation is easy to understand but suffers from being too concise, insufficiently labeled, not covering all relevant matters in sufficient detail, and lacking cohesion between features. For example, there is no index in the FOCUS primer. The examples offered in the user's manual are, as would be expected, of a very general nature, and, therefore, lack any absolute applicability to any one organization's applications. But, there are not enough examples of complex queries. Users felt that the manual could be expanded to at least another volume, perhaps even two more, in order to include more examples. The examples in the basic demo are relatable to actual applications but just are not the same as specialized applications.

5. Vendor Support

Customers have found IBI to be very supportive. System delivery was on schedule for the most part, the systems has performed as expected, and responsiveness to considerable hot-line and help-line traffic has been quick and thorough. Customers have found themselves to be making heavy use of the two lines after initial installation, averaging several calls per week. Hot-line usage lessened dramatically after personnel gained experience with the system, lowering the average to one call every two to three weeks. Customers were impressed with both the speed and the accuracy of the responses received from the help facilities, especially since the inception of the two line concept. All companies stated that they have found "bugs" in the system and that IBI was always responsive to their notification of such problems. They could not be fixed overnight, but users felt that IBI wasted no time in offering at least a temporary fix, while working on a permanent correction. Additionally, users stated that IBI notified other users of the potential problem. It should be pointed out that users were not significantly inconvenienced by these "bugs." As one user stated, "All systems have bugs, it's just a matter of time before you run into them." Indeed, one of the laws of software design, the law of cybernetic entomology, states, "There is always one more bug." The point is that "bugs" do occur but that so far nothing of monumental importance has been affected. The "bugs" can be navigated around or corrected by an in-house fix or vendor correction.

6. Hardware/Support Conversion

FOCUS does require considerable memory, a minimum of 450K for the basic system. All companies interviewed had suitable IBM mainframes (370, 3033, or 4300 series) prior to acquisition of FOCUS, and none required required any hardware conversions or peripheral purchases to employ the FOCUS system. Companies view the FOCUS system as worth the extra outlay of funds. If additional memory or other conversions/support became necessary, they would have no qualms about obtaining it. The alternative of disposing of FOCUS would not be considered.

7. Security/Access

FOCUS operates completely within the security and access controls already in operation in a company's operating system. Additionally, it offers password control, command control, read-write access, and access authorization down to the field level. Customers have expressed satisfaction with the security aspects of the FOCUS system. Several mentioned, however, the questionable security measures inside the organization posed the fact that many users tape their passwords to their terminals so they won't forget them! It was found that the smaller the company was and the fewer people having access to the system, the less concerned management was with security/access aspects of the operating system in general, and of FOCUS in particular.

8. Performance

While no one software tool can be the optimum choice for all people, FOCUS has been well accepted by all levels of workers. The limited main vocabulary and easy understandability of the language make it an acceptable tool for the beginner, while the more complex capabilities allow the more experienced user some flexibility. Thus, beginners are not frustrated with complexities and more advanced users are not limited by simplicity. FOCUS can be as simple or as complex as the user wishes to view it. It is for this reason that organizations feel that the implemetation of the language has met with a minimum of user resistance. In this same vein, the add-on packages, such as graphics or statistical modeling, have given added flexibility to programming applications. Most all of the companies interviewed had purchased at least two of the optional packages, although the evaluations of their usefulness was varied. Some organizations swore by the graphics package, while others had found that they did not have the applications bulk to support its purchase, or, in one case, that the package was just too limited for their particular needs. Similar responses were received on the FIDEL package; some companies rave over it, a few others just don't use it. As is the case with RAMIS II, FOCUS can be operated in both batch and real time environments, and performs reliably in both. Most applications, however, are done in interactive sessions for ad-hoc queries, data entry and retrieval, and report generation, while batch processing is mainly reserved for weekly or monthly functions such as database update and certain summary reports.

9. Improvability/User Satisfaction

Without exception, all organizations interviewed expressed very positive evaluation of the effects on productivity brought about by the use of FOCUS. Except in one case where FOCUS was specifically purchased to replace another DBMS, FOCUS was being used to augment the already existant data processing facilities, albeit to a large degree in some organizations. By far the most widely used host language is COBOL, but many companies also use FORTRAN, PL/1, or SAS. It has already been pointed out that nonprocedural languages can offer significant reductions in time over procedural languages when the applications are applicable. It must be remembered that there are those applications for which nonprocedural languages are not the most efficient or sensible route to take. They have not yet marked the death of traditional procedural languages. There was no user resistance to speak of concerning the implementation of FOCUS. Actually, most of the new "programmers" felt a kind of status associated with using a computer terminal. The actual programmers, of course, had their own terminals, and expectedly found this to be an absolute necessity. Initial hesitance towards using the machine was quickly overcome when the nonprogrammers realized how easy it was going to be, and how quickly they would be able to produce results. In short, user resistance is not a concern.

10. General

FOCUS basic features and optional packages obviously have provided users with enough capabilities from which to choose. None of the users had even considered asking for any tailor-made features. Some users felt that they had more capabilities at the present time than they fully realized and had no need for anything else. No one particular feature of the system could be singled out as more error-prone than any of the others. As mentioned earlier, the more complex the query or application became, the more error-prone it became. This is a function of the application complexity, though, not of the language features themselves. Users are most anxious to see improvements in the quality of the documentation more than any of the features of the language capabilities themselves. Users express satisfaction with the functioning of the system but want a more well-documented library of user information to enable them to reap the benefits of all the capabilities of their system.

11. Overall Assessment

Respondents unanimously felt that FOCUS had caused an improvement in productivity, although none of the respondents had conducted empirical studies on the matter. FOCUS users explained that they simply knew they were getting work done much more quickly than they would have without FOCUS. Calling upon experience with other systems, users compared the expected man hours for job completion against the much lower actual man hours employed using FOCUS, as the determinant of the rise in productivity. Various means of estimation were employed, and varied from company to company. One manager estimated that a FOCUS application which took him one and a half days to complete would have taken 6 months in COBOL. Another manager calculated a 50% rise in the physical output of his division. Yet another stated that he was at the point of needed additional employees to keep up with all the organization's requirements. The introduction of FOCUS alleviated the burden so that additional hiring was not needed. Users also varied in the area of pre-purchase feasibility studies. They fell into basically two categories:

- those who did conduct comparisons
- those who had used FOCUS before and wanted it again

As mentioned earlier, those organizations actually conducting comparisons limited themselves to a very few. Those comprising the second group were the most enthusiastic about FOCUS. They had used FOCUS at another organization and found it to be a system which worked so well for their applications that they felt neither the need to run comparisons with other languages nor to do a feasibility study.

FOCUS features and capabilites are abundant and should be more than adequate for the average user Large users with complex and rigorous requirements pose a challenge to FOCUS capabilities; however, they seem to be meeting it. For example, an industrial conglomerate maintains its corporate data base with FOCUS, Pacific Telephone has over 5000 employees trained and using FOCUS, and Home Box Office (HBO) uses FOCUS to support the majority of its development needs. The two weaknesses in the language are apparently its lack of reentrancy and its inability, until recently, to support concurrent updating. The latter became available with the latest version. However, users did not seem to be troubled by these limitations. In general, FOCUS users seem to be a group rather adamant about the language. The forming of a user's group completely independently of the vendor is an example of such dedication. Bearing in mind that a company is not about to denigrate a product which it has just spent a considerable amount of funds to purchase, the positive reactions of the FOCUS using organizations would be expected. But the users freely expressed their discontent, little though it was, over matters such as the documentation or the inconvenience of system "bugs." All in all, the users are very pleased with their systems and have stated that FOCUS would be their choice if they had to choose again. It would appear that such loyalty has been earned by IBI, both through its effective product which apparently delivers as promised and through the dedication to its customers that it exhibits in its support functions.

VI. COSTS AND BENEFITS ANALYSIS

A. INTRODUCTION

This chapter of the thesis will deal with the analysis of the costs and benefits associated with the nonprocedural product. It will not be a numerical analysis to determine whether an organization with X users, Y programmers, Z systems analysts, and a two and half year programming backlog should acquire a nonprocedural product or hire more programmers. It will not be an attempt to establish the opportunity cost of a nonprocedural product for a particular organization at a particular time. Although periodic inference may be made to cost trends in the data processing arena, there will be no quantifiable comparison or determination made as to whether one should or should not acquire or use a nonprocedural language and its associated software.

Instead this chapter will list and discuss what are generally associated as the costs and benefits of the nonprocedural products from initial considerations to acquire the product to the actual product usage. Some costs and benefits are quantifiable, some are not. Some deal with the social aspects of computer usage, some deal with computer resource management, and still others deal with the theme of this thesis, productivity. From the following anslysis and discussion questions about the positive and negative aspects of the nonprocedural languages will be clarified and answered.

B. COSTS

Companies considering acquisition of a nonprocedural programming language and the software that accompanies it have alot more to think about than the price lists associated with the product. They must also consider the impact on their present system, its effect on resource usage, how the efficiency of the product will additionally effect resource usage; how much if any additional hardware or storage requirements will be needed, or will a new computer be needed; how much training will be needed to achieve the desired level of expertise; will the new system work well with existing computer systems; and while training and implementation is being accomplished, how will it effect the firms operations and the programming backlog. All these items are costs, but through further discussion they will take on a new light and an even newer significance.

As can be seen by the price lists of RAMIS II and FOCUS in Appendices D and G, the direct costs of these systems are quite substantial. Only organizations that can truly benefit from these systems look beyond this point which is indicative of the amount of DP that the acquiring organizations do. These are only the truly visible costs. Although all costs must be considered and taken into account prior to acquisition, many of the costs of the nonprocedural product are not as visible. The following paragraphs will shed more light on this.

One of the first items of cost to consider is the various costs associated with training. As in the case of RAMIS II and FOCUS again, Appendices B and F, training costs are substantial but necessary. Without adequate training, the system goes idle and is useless, incurring additional cost. Even with adequate training the costs of instruction are only part of the cumulative training costs. The users that must sit through the classes are lost for that period of time. The organization must plan for this manpower loss as not to upset operations even further. Companies that plan on conducting their own training must consider the costs of hiring instructors or the loss of manpower that will accompany training their existing staff to accomplish this function. Even after all the classes are over, there is additional training and consultation services that must be accomplished prior to having the system up to full or expected capability.

The next item of cost is the NPL and its associated software, and ultimately its effect on resources. The following excerpts from an article from <u>Datamation</u> should relay the general opinion regarding this usage.

> Computer resource usage is high. A 4GL* uses up to 50% more computer resources than does a 3GL performing an equivalent function.
>
> The computer using a 4GL must have virtual memory and high-speed I/O handling. Fast I/O is essential. ....the major 4GLs are megabyte programs, so virtual memory is mandatory until someone burns one into a ROM. [Ref. 29: p. 116]

It is true, and there is no arguing that the NPLs and their related software use a great amount of computer resources compared to the structured/procedural languages. Some organizations would have to consider acquiring more memory and storage capacity, others would have to acquire

----

* 4GL refers to the 4th Generation Languages, or Nonprocedural Languages, referred to in this paper such as NOMAD, FOCUS, AND RAMIS II.

equipment to increase speed of I/O, and still others would require new systems due to the fact that many NPLs are restricted to certain types of manufacturers and hardware. Many times these drawbacks do not occur or are overcome very easily. Considering the amount of processing that needs to be done to justify acquiring these systems, in the majority of cases, organizations do not need any hardware additions or conversions to support the new system. In regards to the aspect of speed of I/O, managerial control and reservation/assignment of resources can make this problem unnoticeable. An example of this was noted with the use of RAMIS at Citibank, New York.

For some applications, CPU time has been cut by 72% and disc I/O by 84% [Ref. 30: p. 83].

By keeping track of the use of resources, the various application usage, and the volume and periodicity of usage; certain widely-used applications can be permanently put in memory or put on direct access devices to make better use of resources and ensure that overburdening the system does not occur. This is an area that must seriously be considered by the management of the computer installation, and is also dependent upon other computer considerations.

Also in the area of resource usage is the fact that the NPL was/will be acquired to save the time of programmmers. Through the use of the NPL and the software, the computer is doing the work of the programmer.

....we have found out that the increased overhead averages 10% to 30% and is primarily dependent upon transaction volume, access method, and record screening criteria. The root of the problem is actually a major benefit of the 4GL- 4GLs make the computer do alot of the drudge work that the 3GLs make people do. [Ref. 29: p. 116]

Continuing with this view of resource usage as an evaluation of human versus computer efficiency, one must consider the amount of time and resource usage a programmer using structured/procedural methods would normally use. Where a programmer can write a program using an NPL which is written and tested in a matter of days, the programmer using prescribed structural methods will take weeks and possibly months to write and debug a program that does the same thing. The CPU time to execute the numerous simple statements which make up the NPL application will take 9 CPU minutes while the structured program will take 20 CPU minutes to execute a few complex programs and system sort activities. The fact is that the majority of resource usage of NPLs comes from I/O activity, not CPU time, and as previously stated much of that can be eliminated with good computer resource management.

The producers of the NPLs realize the shortcomings and drawbacks of their products and constantly work to correct or improve this area. Through concepts of data independence, elimination of redundancy, and transaction monitoring, the companies try to lessen the impact on computer resources.

The last area of costs of the nonprocedural product lies in the area of acceptance of the product. The users of the product are the people that effect this cost by their acceptance or nonacceptance of the new system. The first type of user to be discussed is the experienced programmer who is assigned to the organization's data processing department. His nonacceptance of the system could be due to the fear of losing his job security due to the increased importance of nonprogrammer users or the possible staff reductions which could occur. Although these fears are generally unfounded due to the fact that these same programmers usually become the trainers and counselors to the nonprogrammer users, it is still a valid consideration as hindrance in this area could have a profound effect on organizational implementation.

The other type of user that could have a noticeable effect on the cost of the NPL acceptance is the nonprogrammer. What they experience is more of a culture shock. Their previous experience with computers is either seeing it or being told not to touch it. They could feel that the organization is forcing it upon them, and they have no real desire to change their job description to include programming duties. Generally the resistance of the programmers and their nonprogrammer counterparts are nonexistent, but they are still valid considerations in successfully implementing a new nonprocedural system.

As seen from the previous paragraphs, the costs of the NPL can be quite substantial in the areas of cash outlays, computer assets, computer resource usage, and the socio-psychological influences. Some are easily identifiable, some are not. Some are easily quantifiable, some are not. In order for organizations to efficiently evaluate the validity of the acquisition of the nonprocedural product, careful evaluation and consideration must be given to these costs in comparison to the expected benefits to arrive at an appropriate decision.

C. BENEFITS

The benefits of the nonprocedural product are well recognized, well published, and widely accepted as one of the few means available to meet future DP demands. As with the discussion of costs, some are quantifiable while others are not. The big difference is their recognizability. Most of the benefits are easily seen, and are the reasons that today's organizations buy the nonprocedural product. These reasons include learnability, user friendliness, increased decision support, more efficient use of human resources, productivity enhancement, adaptability, portability, flexibility, and aid in the description and analysis of these benefits, a chronological approach will be taken as in the discussion of costs, from acquisition to having the system in "full swing."

The first benefit to be seen after acquisition of the NPL is the increased learnability. Due to the simplicity of most NPLs, they are readily learnable and easy to use. Nonprogrammer users can develop enough skill to be producing knowledgeable and meaningful queries and reports in days where procedural languages would take weeks to learn the same skills. This simplicity is also evident in the structure of the language where the user must learn English-like queries to produce the desired results rather than the procedural queries which require strict and more difficult to learn structure and format. The procedural code is also more cumbersome than the nonprocedural code. In some cases NPLs offer a "90% reduction in physical code" over procedural languages [Ref. 29: p. 109]. Even in the case of the more procedural 4GLs, there is much more ease of use and simplicity than 3GLs. Another aspect of the NPLs which tends to increase their learnability is their "user friendliness." Through screen interaction with the user, the NPL can prompt the user toward correction of errors, and lead the user to produce the desired program/application/query more easily. Some NPLs offer advanced interactive data editing features which increases the language's user friendliness and enhances its learnability even more. This degree of helpfulness varies greatly among NPLs and is dependent upon the particular product. Also many of the characteristics which contribute to improved learnability, are primary causes of the other NPL benefits, as will be seen in the succeeding paragraphs.

The improved learnability, simplicity, ease of use, and user friendliness contributes to shorter implementation times. The acquiring companies are able to take advantage of the other benefits of the NPL shortly after installation. There is minimal additional backlog and the additional costs of an idle system are eliminated or kept to an absolute minimum.

At this stage where the system is up and operating and the users are educated enough to accomplish fairly substantial programming functions, the increased decision support aspect of the NPL becomes apparent. Numerous queries, combinations of information, and analyses of data can be accomplished in a fraction of the time that a procedural language would take. The flexibility of the NPLs also contributes to this aspect as changes can easily be made to nonprocedural requests to make the program meet exact specifications, while similar changes to procedural language programs would probably require additional systems analysis and major rewrite. Read and Harmon comment directly on this in their article "Assuring MIS Success." They are talking about the 4GL's ability to do this iterative requirements analysis.

Simply stated, define the detailed requirements, program the system. Show it to the user; if it's not right, repeat this cycle again and again until it is right. If we were using COBOL programs in dealing with large systems, they would become unmanageable after the third iteration. With a 4GL's flexibility, corrections are easy. [Ref. 29: p. 116]

Also in the area of improved decision support is the speed of access to information. The tremendous ad-hoc capabilities of the nonprocedural products puts all necessary information at the user's/manager's fingertips, and provides for, reinforces, and supports valid decision making. The user can get the required information in time to use it.

Another area to be examined is the NPL's or 4GL's effect on productivity. This productivity enhancement can be expressed in time savings. It can be expressed in increased programmer productivity or the increase of the user's ability to generate more queries, reports, applications, and systems in less time as compared to procedural languages. It can also be expressed in the resultant profit of the organization due to the increased decision support of the nonprocedural product. The NPL and its associated software aids in the generation of applications by nonprogrammers. Although most of these applications are routine and noncritical, they make up the majority of applications and take up the majority of programmer time to produce. By allowing the nonprogrammer users to produce these applications, the NPLs free the experienced programmers and system analysts, who would normally be doing this programming, to do their respective jobs of programming and analyzing. The programmer becomes free to attack the critical applications for the organization and reduce the companies application backlog while the analyst is able to devote his efforts to design and analysis.

Due to their simplicity and their reduction of physical code, NPLs allow for production of smaller, less complex, and easier to understand applications. According to James Martin this also contributes to programmer output/ productivity.

The comparison of productivity with large and small programs indicates that development of large programs should be avoided by DP departments wherever possible [Ref. 1: p. 41].

This reduction in size and complexity is also beneficial in reducing the costs of program correction or "debugging", which can amount to as much as "20% of total cost with small programs" and "50% with large programs" [Ref. 1: p. 41]. The noticeable increases in productivity are due to the fact that the software that has been and is being developed with the nonprocedural systems does much of the work that the programmer used to do. The ability for productivity enhancement by the nonprocedural products has been commented on and substantiated by many authors. A few are stated as follows.

...the productivity increases...are available now, and the gap (the productivity gap between 3GLs and 4GLs) will widen as the 4GLs develop.

Using a powerful 4GL, combined with software factory methods, provides a quantum leap in both information control capability and programmer productivity. It also opens up programming to a much larger section of the work force. [Ref. 29: p. 120]

In stark contrast to the surveys of programmer productivity improvement are the results that have been achieved with data-base user languages, report generators, graphics packages, and application generators. With these, productivity improvements of over 1000% are not uncommon [Ref. 1: p. 44].

The last benefits to be discussed deal with the vendor of the NPL, and are some of the main considerations in the cost/benefit analysis. These benefits are the adaptability and portability of the 4GL. With the vast steps being made in computer technology, shifting an organization from one computing environment to another could mean massive environmental maintenance, modification of programs, and retraining of users. By providing this adaptability or portability, the vendor offers a commitment to the client organization that changes to his hardware or software environments will not effect the applications and user requests that allow the organization to function. The only apparent limitations on this portability is the current practice of the vendors limiting usage of their products to certain hardware manufacturers. This adaptability and portability must be a serious inclusion in any organizations future DP plans.

Portability is essential if an organization is to avoid excessive conversion costs and reap the benefits of new technology [Ref. 31: p. 34].

D. CONCLUSION

The present and future demands for computer applications are tremendous. In order to meet industry's DP needs will require a shift from present procedural methods. Only through the growth of the nonprocedural products and the accompanied software to enhance programming abilities can the needs and demands be met. Of course consideration must be given to the inherent costs of the systems, but with today's computing industry marked by the rising cost of human resources, decreasing hardware costs, an overburdening demand for applications, a shortage of programmers to produce those applications, and the constantly changing computer environments, there is no doubt that the benefits far outweigh the costs. The organizations that consider the acquisition of an NPL must not only conduct serious analysis of the benefits to be received or the costs avoided by the many various products offered, they must also analyze the comparative degree to which these products and their respective vendors commit themselves to provide for future DP requirements.

VII. CONCLUSIONS

The results of the research into nonprocedural languages has led the authors to independently arrive at the same conclusions concerning usage and productivity. Although the numerical values of the statistics gathered by each author are not equivalent, they are a rough approximation of the results that can be expected from the use of nonprocedural languages in general. Comprehensive, overall ratings of the languages by their users are listed in Table 7.

First, and most obvious, is the tradeoff between human and computer resources. Nonprocedural languages offer a quantum leap in application productivity as compared to structured programming methods, but they do this at the sacrifice of computer resource efficiency. Although this inefficiency must be considered as a cost concern of prospective buyers, the following considerations must also be taken into account.

1. Much of the computer resource usage is not due to inefficiency, but is due to the fact that the NPL's software and the computer are doing the work that the programmer would normally do using structural methods.
2. Computer costs and the cost of computer time is falling while programmer costs are rising. The following quotation is applicable.

> costs of computer time and people time are changing. ...Before long the cost of a person for an hour will be ten times greater than the cost of a computer for an hour [Ref. 1: p. 3].

3. There is not enough programmers to meet the growing demand for applications using structured procedural methods.

TABLE V

Overall Assessment of Nonprocedural Languages

	Excellent	Good	Adequate	Poor
Implementation	4	14	3	0
Training	10	11	0	0
Learnability	8	11	2	0
Documentation	11**	7	3	0
Vendor Support	7	13	1	0
Hardware/Support Conversion	19	2	0	0
Security/Access	2	4	15	0
Performance	17	4	0	0
Improvability/ User Satisfaction	15	6	0	0
Overall Assessment	16	5	0	0

** As mentioned earlier, the quality and content of the documentation is excellent; however, user complaints over the accessiblity and physical layout of the information indicates that there is room for improvement.

Computer resource efficiency is important, but is becoming the subject of increasingly less concern. It is obvious from a review of the price lists in Appendices D and G that the nonprocedural languages themselves do not come cheaply. The acquisition of the nonprocedural product is an internal decision that each organization must make, weighing programmer time, salaries, project priorities, availability of both programmers and nonprogrammers, computer capacity and costs, and training time and expenses.

Secondly, although largely aimed at the nontechnical user or nonprogrammer, many of the experienced programmers in the organizations are actually the first to use the nonprocedural languages. With their newly developed skills, the programmers become the resident experts or counselors, and can hopefully avoid or more easily correct any problems encountered by the nonprogrammer users. This practice is debatable as many experts claim that the nonprogrammer has the advantage over programmers in the use of the NPLs.

The most efficient user of the nonprocedural languages is one who has never written COBOL [Ref. 32: p. 12].

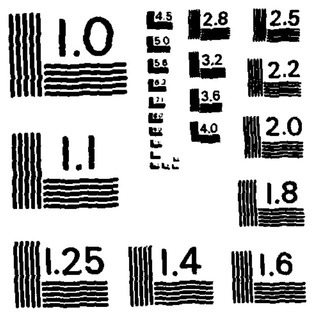
Despite this belief, it is felt that experienced programmers have a better understanding of what the language is doing, and can more easily grasp the idiosyncracies of the language that the nonprogrammer would not pick up. Although the NPLs were clearly designed for nonprogrammers to use for simple data entry and retrieval, updating, ad hoc queries, and report generation, which makes up the majority of data processing, the practice of limiting nonprogrammer usage until the programmers and analysts have a grip on it, appears to have a sound basis.

The third item is the expectations that organizations have of of the NPLs. Because of the simplicity and ease of use of the NPL, many expect overnight results. They lose sight of the need for complete training and the value of user experience. The expectation of premature benefits often times leads to user frustration. Also overlooked is the fact that to construct increasingly complex programs requires as much if not more knowledge of the language than a 3GL would require.

AD-A127 754 A PRODUCTIVITY ANALYSIS OF NONPROCEDURAL LANGUAGES(U) 2/2
NAVAL POSTGRADUATE SCHOOL MONTEREY CA
M CORCORAN ET AL. DEC 82

UNCLASSIFIED F/G 9/2 NL

END
DATE
FILMED
5 83
DTIC



With current 4GLs, programming complexity rises exponentially with product complexity, and to be functional at the upper levels requires a considerable amount of knowledge and experience [Ref. 29: p. 120].

Despite the expectations that the NPLs will provide overnight sensations, there is a "learning curve" that must be taken into account. The benefits of the system may be realized fairly rapidly, but they are not instantaneous.

Fourth, users of the nonprocedural languages agree that their productivity has improved because of the language usage. The degree of improvement experienced is as variable as the reasons for basing the increases in productivity. In fact, exact figures on how much improvement has occurred is impossible to calculate since companies have not collected any empirical data to form judgements as to the actual improvement made. Instead they have relied upon subjective managerial observations and estimations.

Fifth and last, there are applications for which nonprocedural languages are not the best choice.

Computation intensive work is not handled well (by 4GLs)...Scientific 3GLs such as FORTRAN should still be used for heavy computational work. If an application requires both character-crunching and number-crunching, write it in a 4GL, but call the 3GL for computational work. [Ref. 29: p. 116]

Nonprocedural languages are a powerful and extremely useful alternative to much of today's applications programming, but they cannot, at this stage, completely replace third generation languages.

Fourth generation applications development systems aim to fully integrate all user services in a user friendly software product which is easy to learn and use for the new users and efficient and complete for the skilled professional. These applications development systems are the trend for the future. Through their various and numerous capabilities and the vendors' commitment to adapt the products to fit technological and customer growth, the nonprocedural languages can be extremely valuable productivity enhancment tools for those organizations in positions to justify their cost.

APPENDIX A
RAMIS II USERS

Monsanto Company

Citibank

Lockheed Missile and Space

Insurance Corporation of British Columbia

Southern Pacific Communications

Advanced Micrcdevices

Federal Home Loan Bank of San Francisco

Litton Mellonics

APPENDIX B
RAMIS II TRAINING COSTS

Training Prices/Fees

Student days/year	Cost/day
200+	$70
151-200	80
101-150	90
51-100	100
1-50	110

Organizations can contract in advance with RAMIS II for reduced rates.

On-site courses at customers location is $1,100/day. The cost for MPGSWIFT Courses is $600/day.

APPENDIX C

RAMIS II COURSES OFFERED

Preview of RAMIS II Release 2.2 (newest version)
Basic RAMIS II Reporting (for nonprogrammers)
Basic RAMIS II Reporting
Basic RAMIS II File Design and Records Management
Advanced RAMIS II File Design and RecordsManagement
Advanced RAMIS II Reporting
RAMIS II Designer's Workshop
Using RAMIS II Efficiently
Advanced Reporting Options
RAMIS II Executive
RAMIS II Financial Planning Option
Using Procedural Languages with RAMIS II
Describing IMS Files to RAMIS II
MPGSWIFT Applications Programming (Advanced RAMIS II)
MPGSWIFT Systems Programming

APPENDIX D
RAMIS II PACKAGE PRICES

Description	Single Payment	Monthly Payment Annual Rental
RAMIS II Basic Package NPL for Report Preparation and Records Management		
For a Large CPU (Model 65,155,4341 and larger)	$48,000	$1250
For a Medium CPU (Model 50,148,4331 and smaller)	24,000	625
Procedural Language Interface	15,500	405
Communications Interface (CICS,ETSSII,GUTS,ICCF,IMS/DC, ROSCOE/RTS,TSO,VM/CMS)	14,500(EACH)	380(EACH)
Operating Systems Interface (DOS/VS,DOS/VSE,MVS,OS,VM/CMS, VS)	5,500(EACH)	145(EACH)
High Resolution Graphics	8,500	225
Usage Accounting	2,500	65
Financial Planning(DFO)	7,500	195
GRAPH	3,500	95
Executive	10,000	260
APL Interface	8,000	210
Interactive Request Modification	NO CHARGE	NO CHARGE
IBM/Extended	3,500	95
External File Interface	6,500	170
Automatic Interface (to ADABAS,DL/1,IDMS, IMS,TOTAL)	6,500(EACH)	170(EACH)
Read OS/DOS	6,500	170

APPENDIX E
FOCUS USERS

City of Fresno, California

American President Shipping Lines

Holbrook Enterprises

UCLA Computer Center

Blue Cross of Washington and Alaska

ESL Incorporated

First Interstate Services

Guy F. Atkinson Company

Lamb-Weston, Incorporated

LDS Church

Rolm Corporation

Santa Fe International Corporation

Syntex Incorporated

APPENDIX F

FOCUS TRAINING COURSES AND FEE SCHEDULE

Course 101	File Description and Maintenance	1 Day	$125
Course 103	Basic Report Preparation	2 Days	$265
Course 105	Timesharing Workshop	1 Day	$125
Course 107	Basic Report Preparation	1 Day	$ 60*
Course 102	Basic Report Preparation	2 Days	$265
Course 104	File Description and Maintenance	1 Day	$125
Course 106	Timesharing Workshop	1 Day	$125
Course 108	Basic Report Preparation For Managers	1 Day	$ 60*
Course 201	Advanced Techniques Workshop	2 Days	$250
Course 202	What's New in FOCUS	1 Day	$125
Course 203	FOCUS Internals	2 Days	$380
Course 301	Decision Support	1 Day	$125
Course 302	Host Language Interface	1 Day	$125

Course 101/104 provides the basic training needed to create and maintain FOCUS databases. Single path, multi-path and network structures, file maintenance facilities and techniques are covered.

Course 102/103 covers the basic elements of report preparation requests, including data retrieval, sorting, record screening, format control and calculations.

Course 105/106 provides a workshop utilizing basic FOCUS and timesharing commands to give new FOCUS users hands on experience within the CMS and TSO environments.

Course 107/108 gives comprehensive overview of FOCUS reporting capabilities including FOCUS graphics, report formatting, data retrieval and calculations.

Course 201 provides a problem solving workshop using advancedFOCUS techn techniques for building applications on data types, their relationships, and reports to be produced. These techniques and concepts not readily available in the FOCUS Users Manual.

Course 202 is designed to provide experienced FOCUS users with a periodic update of the latest enhancements to FOCUS.

Course 203 is designed to give experienced FOCUS users insight into the essential internal operations of FOCUS, including structures, relationship of segments, use and maintenance of indexes, sorting, and logical vs. physical file traversals.

Course 301 presents the advanced features of FOCUS reporting, dialogue control graphics, statistical capabilities and their use as a decision support tool.

Courses 101, 103, 105 and 107 are designed for experienced computer users, approximately one year programming experience; Course 102, 104, 106 and 108 are designed for new computer users. Both sequences are intended to be taken sequentially. The remainder of the courses are designed for users with a working knowledge of FOCUS.

*Based on minimum charge of $900 for first fifteen students. Course offered on site only.

APPENDIX G

FOCUS PACKAGE PRICES

INFORMATION BUILDERS, INC.

VERSIONS
VM/370 CMS
TSO/OS/VS/MVS/CICS

February 1, 1982

FOCUS FEE SCHEDULE

ITEM	Onetime License	Monthly License
BASIC SYSTEM		
FOCUS Report Generator and Dialogue Manager for Reporting from FOCUS and/or external files	$43,000	$1,170
FOCUS Data Management, Transaction Processor and Interactive File Scanner	$23,000	$510
OPTIONAL FEATURES		
FOCUS Host Language Interface	$8,500	$210
FOCUS Statistical Analysis Package	$6,500	$180
FOCUS Graph Subsystem	$8,500	$210
Modelling Language for Financial Reports	$8,500	$240
FIDEL (CRT Data Entry Language)	$5,500	$150
TED (Tiny Editor) for Editing from within FOCUS	$2,000	$45
Central Data Base Control for Simultaneous Users	$8,500	$240
CP/Assist Installation Option	$2,500	$65
FOCUS/APL (VS/APL use of FOCUS files)	$6,000	$250
OPTIONAL DATA INTERFACES		
IMS Interface to report from IMS files	$8,500	$240
IDMS Interface to report from IDMS files	$8,500	$240
TOTAL Interface to report from TOTAL files	$8,500	$240
ADABAS Interface to report from ADABAS files	$8,500	$240
OPTIONAL COMMUNICATIONS INTERFACES		
CICS Interfaces for interactive operation of FOCUS under CICS	$8,500	$240
IMS/DC Interface for interactive operation of FOCUS under IMS/DC	$8,500	$240
FOCUS/CMS to FOCUS/OS Bridge	$2,000	$60

INFORMATION BUILDERS, INC.

FOCUS FEE SCHEDULE February 1, 1982

TRAINING

- With the monthly license, an application specialist is provided for three days to conduct a training program.

  With the onetime license, an application specialist is provided for ten days for both application consulting and training.

- Regularly scheduled courses conducted at IBI sites are $125 per student per day.
- Additional education conducted on the customer's premises may be obtained at the rate of $900 per day for up to 15 attendees plus $30 per additional attendee over 15.
- Out of pocket expenses are charged as incurred for onsite support and education.

USER MANUALS

- With the monthly license, five manuals are provided at no cost.
  Ten User Manuals are provided free with the onetime license.
  Additional User's Manuals are $9.00 per set plus shipping.
  Query Language Primers are $6 plus postage.
  Quick Reference Guides are $2.50 each (ordered in quantities of 5).

ANNUAL ENHANCEMENTS AND MAINTENANCE

- The Onetime License fee includes the first year of maintenance and enhancements. After the first year, enhancements and maintenance are optional and are 10% of the current onetime fee.
- The Monthly License fee includes all enhancements and maintenance.

MONTHLY LICENSE CONVERSION

- One half of the most recent 12 Monthly License fees paid can be applied towards a Onetime License.

USAGE

- The in-house license of FOCUS is for use at computer centers either wholly owned by Licensee or at least 50% owned by Licensee.

ADDITIONAL SYSTEMS

- Additional FOCUS systems for use on more than one CPU in North America can be obtained at a reduction from the single CPU license fee if obtained within a 24 month period.
  2nd system — 50% of single CPU fee
  3rd system or more — 40% of single CPU fee

MONTHLY LICENSE PERIOD

- The Monthly License is for a one year period, but may be cancelled by the Licensee at any time upon 30 days notice.

THREE MONTH TRIAL PERIOD

- Monthly License fees are charged for 3 months. If in the fourth month the system is purchased, 100% of the trial period fees are applied towards purchase.

APPENDIX H
RESEARCH QUESTIONAIRE

1. Implementation

   a) When was the system installed?

   b) Why was this one chosen?

   c) What others did you consider?

   d) How many people use it?

   e) How many can use it at the same time?

   f) How many people in the total organization?

   g) What is its main use? (Real time, analysis, model building, etc.)

   h) Is processing done for any subsidiaries/dependent organizations? If so, what is the nature of the processing?

   i) What percentage of information processing is accomplished using FOCUS/RAMIS II?

   j) How long did it take you to get into full swing?

   k) What level(s) in the organization use it?

   l) Did you ultimately require less manpower?

2. Training

   a) What training was done?

   b) How much time was devoted to it?

   c) Was it vendor provided?

d) Did you choose a blanket or per person rate?

e) How many people were trained?

f) What follow-up and/or refresher training is done?

g) What about new beginners?
(in-house or vendor trained)

h) What future training plans do you have?

i) What manuals, resident experts are readily available to the users?

j) Was there any work backlog while in the startup phase?

k) Was there a need for temporary hiring while in the start-up phase?

   i) How many?

   ii) How long were they employed?

   iii) What were they doing?

3. Learnability

a) Have you found high retainability?

b) Were vendor estimates of training time accurate?
Off by how much?

c) Do user errors present a significant problem?

4. Documentation

a) How do you rate the quality of vendor documentation?

b) Is the documentation easy to understand?

c) Are all relevant matters covered?

d) How about user complaints over the documentation?

e) Do the users refer to the documentation when they need help?

5. <u>Vendor Support</u>

a) Did the vendor deliver on time?

b) Has the system performed as expected?

c) Vendor hot line

i) Have you used it?

ii) How often?

iii) What has been the speed of the response?

iv) What has been the accuracy of the response?

6. <u>Hardware Support/Conversion</u>

a) Did you require any extra hardware to support the system?

b) Did you require more memory? How much storage does it use?

c) Were any hardware conversions necessary?

d) What host languages do you employ?

7. <u>Security/Access</u>

a) What security measures are in effect to control access?

b) How is data protected from unauthorized change/loss ?

c) Who can change the database?

d) Backup
i) What backup measures do you use?

ii) How often is the backup updated?

iii) How often has it been necessary to use the backup?

8. <u>Performance</u>

a) What changes are possible and are they visible to the user?

b) How well can the language control the computer?

c) How "structured" is it?

d) Library

i) How large is the library?

ii) How useful is it?

iii) Do you expect to add to it?

iv) How much and how fast?

e) Do you write your own programs or are they canned?

f) Do you use a timesharing, multiprogramming environment?

g) Do you use batch or realtime processing or both?

h) What is the system's reliability?

i) Did you need to do any debugging?

j) Do you use static or dynamic memory?

k) Do you have programmers working for you?

9. <u>Improvability/User Satisfaction</u>

a) Would you say that it is an improvement over your previous system?

b) Did you experience any significant user resistance?

10. General

a) Did you receive any tailor-made features?

b) Did you purchase any options? Which ones?

c) Have you found them useful?

d) Have you found any features to be useless or less usable than you had anticipated? Which ones?

e) What are the most used features?

f) What are the most error prone features?

g) What improvements would you like to see?

h) Have you experienced any problems with maintenance of the system?

11. Overall Assessment

a) Has productivity improved?

b) What performance measures have you used to deduce this?

c) Does the system meet/fall short/exceed your organizations requirements?

d) Were any feasibility studies conducted, including cost/benefits analyses, prior to acquisition?

12. Any additional comments?

LIST OF REFERENCES

1. Martin, James, *Application Development Without Programmers*, Prentice-Hall, Inc., Englewood Cliffs, N. J., 1982.

2. Schlaline, E. A., "Computer Programming Languages of Use in Manufacturing," *Software for Computer Systems*, College Readings, Inc., Arlington, Va., 1970, pp. 123-125.

3. Sammet, Jean E., *Programming Languages: History and Fundamentals*, Prentice-Hall, Inc., Englewood Cliffs, N.J., 1969, pp. 19-26.

4. Tagg, R. M., "Query Languages", *Database Journal*, Vol. 11, No. 3, 1981, pp. 7-15.

5. Robinson, M. A., "A Review of Data Base Query Languages," *The Australian Computer Journal*, Vol. 13, No. 4, Nov. 1981, pp. 143-159.

6. Kaufman, Richard L., Goetz, Martin A., and Rin, N. Adam, "Integrated Fourth Generation Software Languages", *Computerworld*, Vol. XVI, No. 35a, 1 September 1982, pp. 37-44.

7. Reisner, Phyliss, "Human Factors Studies of Database Query Languages: A Survey and Assessment," *Computing Surveys*, Vol. 13, No. 1, March 1981, pp. 13-31.

8. Holtz, D. H., "A Nonprocedural Language for On-line Applications," *Datamation*, April 1979, pp. 167-176.

9. Hayes, Phil, and Ball, Eugene, and Reddy, Raj, "Breaking the Man-Machine Communication Barrier," *Computer*, March 1981, pp. 19-29.

10. Myers, Ware, "Computer Graphics: Reaching the User," *Computer*, March 1981, pp. 7-17.

11. Robinson, Tim, "Generators Smooth End-User Programming," *Computerworld*, 30 March 1981, pp. 53-54.

12. Shneiderman, Ben, *Software Psychology: Human Factors in Computer and Information Systems*, Winthrop Publishers, Inc., Cambridge, Mass., 1980, pp. 206-213.

13. Hussain, Donna, and Hussain, K. M., Information Processing Systems for Management, Richard D. Irwin, Inc., 1981, pp. 42-70.

14. Montgomery, C. A., "Is Natural Language an Unnatural Query Language?," Proceedings National ACM Conference, New York, 1972, p. 1075.

15. Watson, Richard, "English '/* is English - Sometimes," Systems User, Vol. 2, No. 10, January 1982, pp. 1-4.

16. Smith, Robert A., "Three Factors Top Productivity Considerations," Computerworld, 30 March 1981, pp. 27-28.

17. Wilson, Peter B., "User Acceptance Plan Keeps Productivity High," Computerworld, 30 March 1981, pp. 42-44.

18. Moynihan, John A., "What Users Want," Datamation, Vol. 28, No. 4, April 1982, pp. 116-118.

19. Hopper, K., User Oriented Command Language, Heyden & Sons, Ltd., London 1981, pp. 1-4.

20. Synnott, William, and Gruber, William, "The Care and Feeding of Users," Datamation, Vol. 28, No. 3, March 1982, pp. 191-204.

21. Tapscot, Don, "Investigating the Electronic Office," Datamation, Vol. 28, No. 3, March 1982, pp. 130-138.

22. Welty, Charles, and Stemple, Charles W., "Human Factors Comparison of a Procedural and a Nonprocedural Query Language," ACM Transactions on Database Systems, Vol. 6, No. 4, December 1981, pp. 626-648.

23. Gray J., "Some Comments on Research Development in Database Management," Research Directions in Software Technology, The MIT Press, Cambridge, Mass., 1979, pp. 807-809.

24. Brooks, Fred P., Jr., The Mythical Man-Month, Addison-Wesley Publishing Co., Reading, Mass., 1979, pp.13-26.

25. Keston, Robert, "Five Obstacles Hinder Performance Measures," Computerworld, 30 March 1981, pp. 21-22.

26. McCormick, Ernest J., _Human Factors in Engineering Design_, McGraw-Hill Book Co., New York, 1976, pp. 457-471.

27. Inmon, W. H., "Study Assessed Query Language Resource Use," _Computerworld_, 31 July 1978, pp. S11-S14.

28. Ciampi, Peter, "End User Languages: Why Slow Acceptance?," _Computerworld_, 30 November 1981, pp. 38-39.

29. Read, Nigel S., Harmon, Douglas L., "Assuring MIS Success," _Datamation_, February 1981, pp. 109-120.

30. Roach, R.E., "RAMIS at Citibank," _Datamation_, December 1976, pp. 83-90.

31. Fish, Frank, "Adapt and Survive-Industry's Tall Order," _Computing_, August 1981, pp. 30-34.

32. "Gloom and Glad Tidings from James Martin," _Computer Decisions_, January 1981, pp. 10-12.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virgina 22314	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93940	2
3.	Department Chairman, Code 59 Department of Administrative Sciences Naval Postgraduate School Monterey, California 93940	1
4.	Curricular Office, Code 37 Computer Technology Naval Postgraduate School Monterey, California 93940	1
5.	Professor Norman R. Lyons Code 54Lb Department of Administrative Sciences Naval Postgraduate School Monterey, California 93940	3
6.	Lieutenant Commander John R. Hayes, SC, USN Code 54Ht Department of Administrative Sciences Naval Postgraduate School Monterey, California 93940	1
7.	Lieutenant Mimi Corcoran, USN HQEUCOM APO New York 09128	3
8.	Lieutenant Denham B. MacMillan, USN SWOSCOLCOM Naval Education and Training Center Newport, Rhode Island 02840	1
9.	Ms. J. M. Corcoran P. O. Box 5221 Redondo, Washington 98054	1
10.	Mr. and Mrs. James E. Corcoran 429 Shogun Drive Greensburg, Pennsylvania 15601	1
11.	Mr. Kit Reichow Lockheed Missile and Space 1111 Lockheed Way Sunnyvale, California 94086	1
12.	Mr. Bob Leclerc Information Builders, Inc. 181 Lytton Avenue Palo Alto, California 93041	1

13.	Mr. Bob Stubbs Advanced Microdevices 898 Stewart Sunnyvale, California 94086	1
14.	Mr. Eddy Rawlings Mathematica Products Group 1100 Larkspur Landing Circle Larkspur, California 94939	1
15.	Mr. Mark Weisler Southern Pacific Communications 839 Mitten Rd. Suite 127 Sunnyvale, California 94086	1

ATE
LME
8